CO-EVOLVING TREES AND CONTROL POLICIES: BRIDGING POSTERIOR OPTIMALITY AND ONLINE EFFICIENCY IN SPECULATIVE DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Tree-based speculative decoding accelerates LLM inference through parallel verification of candidate continuations. Posterior verification records and a calibrated latency model enable statistically optimized tree portfolios with near-optimal surrogate throughput in a finite candidate space. Pre-drafting routing, however, commits an entire cycle with limited information, making mistaken template choices costly and hindering realization of posterior gains. Our preliminary experiments identify two advantages of decisions during drafting: richer observations and reuse of shared computation across alternative actions. We therefore formulate an ideal joint objective for draft structures and online policies: expected committed tokens per expected complete-cycle time. We propose Oracle-Guided Structure–Policy Optimization (OGSPO), using a common net-gain objective for coupled conditional updates. It first solves for posterior-optimal decision depths using verification records and modeled costs in a mixed-integer fractional program, then alternates policy and structure updates. With the draft structure fixed, posterior records and a latency model estimate every legal terminal action's net gain for policy learning. Sampled same-trace executions correct policy-improvement estimates. With the policy response fixed, state-conditioned reach and stop probabilities weight mixed-integer optimization of the shared nested draft structure. We establish conditional unbiasedness of policy-improvement estimates under the sampling protocol and finite-domain optimality bounds for the structural proxy under a fixed response. Across twelve configurations spanning three models, four devices, and two temperatures, OGSPO improves four-workload mean complete-request throughput over EAGLE-3 by 9.2% on average and up to 19.3%. Comparisons and ablations further support the benefits of joint structure–policy design.

1 INTRODUCTION

Speculative decoding accelerates autoregressive generation by letting a small draft model propose candidate tokens for parallel verification by a large target model, while preserving the target distribution (Leviathan et al., 2023; Chen et al., 2023). Tree-based methods organize candidates into multiple shared-prefix continuations, increasing the chance of committing more tokens in each verification step (Miao et al., 2024; Li et al., 2024; 2025). However, larger trees also incur higher drafting and verification costs, so the most efficient tree depends on both the decoding state and execution cost.

Existing methods have largely improved either *tree construction* or *online control*. EAGLE-2, OPT-Tree, and Sequoia optimize candidate expansion or reusable tree structures using draft confidence, expected acceptance, and hardware costs (Li et al., 2024; Wang et al., 2025; Chen et al., 2024). SpecDec++ and Learning to Draft (LTD), in contrast, adapt when to stop drafting and how much to verify (Huang et al., 2025; Zhang et al., 2026a). These two problems are inherently coupled: a controller can only act on candidates exposed by the tree, while the value of a tree depends on when and in which states the controller chooses to use it.

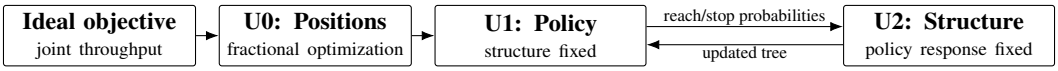
Posterior verification records favor different trees across decoding states: choosing each state's best tree in hindsight makes a small portfolio substantially better than one shared tree. Realizing this gain requires choosing online. On unseen test states, our best pre-drafting selector recovered only 2.0% of the posterior gain, whereas access to verification outcomes raised this to 50-76% (Section 3.2). Realizing structural gains thus requires identifying when to use each tree from online information.

Drafting reveals new candidate scores while supplying a computed prefix that later actions can reuse. Waiting can therefore improve decision information, at additional drafting cost. The question is when

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

this information justifies its cost and how the structure should support such decisions. We therefore optimize the draft structure and its controller together under a common objective: maximizing expected committed tokens per unit of complete cycle time.

We address these questions with Oracle-Guided Structure-Policy Optimization (OGSPO). Starting from the ideal joint objective, **U0** solves for shared decision positions by maximizing the posterior ratio of expected committed progress to expected complete execution time. At these positions, two coupled updates optimize the draft structure and its policy: **U1** optimizes the policy on a fixed draft structure using posterior action net-gain estimates and same-trace residual correction of policy-improvement estimates; **U2** optimizes the shared structure under the learned policy's state-conditioned reach and stop-action probabilities.



After each structural update, action values are recomputed and the policy reoptimized under the common progress–time objective. Independent gain comparisons govern acceptance; gain bounds, stability, or budget govern termination.

Our contributions are as follows:

(i) *From posterior opportunity to **online control**.* We identify the information bottleneck in realizing posterior structural gains, jointly formulate structure and within-cycle control under a progress-time objective, and characterize when later drafting information justifies its execution cost.

(ii) *From sparse feedback to **full-action policy optimization**.* For a fixed draft structure, we use target-side posterior information to value every legal stopping and verification action, rather than learning from only the executed action. Sparse same-trace executions yield conditionally unbiased policy-improvement estimates even when the action net-gain estimates are biased.

(iii) *From policy-independent trees to **policy-guided structure design**.* We optimize the shared structure using state-dependent policy reach and stop-action probabilities, allocating limited drafting and verification capacity while accounting for competition over verification slots.

Across 12 model-device-temperature settings, one structure-policy feedback round improves four-workload mean complete-request throughput by 4.8–19.3% over EAGLE-3 and 2.0–8.9% over LTD.

2 Related Work

Draft-tree design. EAGLE-2 and EAGLE-3 expand candidates by draft confidence during decoding (Li et al., 2024; 2025); Group Tree Optimization aligns drafter training with target-side tree acceptance (Hu et al., 2026); OPT-Tree formalizes the resulting probability-ranked choice as expected acceptance optimization under a node budget and adds a cost-related stopping threshold (Wang et al., 2025). Sequoia moves from per-state expansion to reusable topologies, using sibling-rank acceptance statistics and dynamic programming under a node budget before selecting hardware-aware depth and size (Chen et al., 2024). Hydra likewise studies nested proposal trees from corpus acceptance gains and measured throughput (Ankner et al., 2024). For block-diffusion drafting, CaDDTree optimizes tree and budget design for throughput, DominoTree scores candidate paths conditionally during expansion, and DARTree extends autoregressive correction from chains to trees (Zhang et al., 2026b; Lin & Jang, 2026; Li et al., 2026). SMART grows the current tree with a training-free marginal-gain-to-latency rule (Wang & Zhou, 2026); Yggdrasil combines context-aware tree selection, latency profiling, and a compiler-friendly runtime (Guan et al., 2025). These scores and cost models increasingly reflect execution, while reusable topologies remain optimized separately from stopping and candidate-retention policies.

Online decisions. AdaEAGLE predicts draft length before drafting (Zhang et al., 2024); SpecDec++ and AdaEDL use predicted rejection and draft-logit entropy to stop drafting (Huang et al., 2025; Agrawal et al., 2024); LTD learns depth and verification-size decisions from intermediate draft features using complete-cycle throughput as reward (Zhang et al., 2026a). HeteroSpec adapts drafting depth and pruning thresholds to verification complexity (Liu et al., 2026b); LinguaSpec uses linguistic priors to adapt drafting and verification (Wang et al., 2026). EVICT chooses a cost-effective verification prefix after drafting for mixture-of-experts targets (Pan et al., 2026). LTD can respond to information arriving during generation, but its candidate topology remains determined by EAGLE-3's probability-ranked expansion. SMART instead changes the current tree through a fixed marginal rule. Online choices depend on the generated candidates, and topology value depends on their actual use.

Target-side feedback. Recorded target behavior can evaluate options beyond an executed decision. Dynamic Delayed Tree Expansion trains a root-position selector using offline-valued settings and latency estimates (Thomas et al., 2026); Speculative Verification selects verification length from fitted acceptance and latency signals (Kim et al., 2026); HedgeSpec obtains full-information feedback on candidate drafters from a shared target trajectory (Liu et al., 2026a). These studies use target feedback within a specified decision interface.

Our approach. OGSPO jointly designs the draft structure's observations and terminal actions and the policy that realizes their value, under one progress–time objective: posterior return estimates guide decision-position choices, and policy reach and stop probabilities weight the compilation of a shared nested draft structure. Re-adaptation and complete-execution comparison then evaluate the structure–policy pair that can actually be deployed.

3 Problem Setup

3.1 Draft–Verify Cycles and Throughput

Speculative decoding uses a small draft model to propose candidates and a target model to verify them in parallel, allowing one cycle to commit several tokens. Tree-based speculative decoding methods (e.g., EAGLE-3) arrange these candidates as a token tree. A cycle begins at state S, which contains the confirmed prefix, key-value caches, a pending root token o, and the available context. Drafting produces a tree D and trace ω; each node u has token x_u, prefix h_u, and root-to-node path $\mathrm{path}(u)$. Siblings are ordered by draft probability. Tree attention lets the target score their prefixes in one forward pass, after which verification commits the longest accepted path and determines the next root.

The verifier may use a legal subtree $C \subseteq D$ that retains each selected node's ancestors and higher-ranked siblings. Ordered residual verification preserves the target distribution over retained prefixes; its conditional acceptance rule and stopping conventions are given in Appendix C. Let L count committed non-root draft tokens, C° the non-root nodes eligible for commitment, and P_u the probability of accepting the path to u. Expected committed progress is

$$\begin{aligned}\bar{G}(S,\omega,C) = \mathbb{E}[G \mid S,\omega,C] = 1 + \mathbb{E}[L \mid S,\omega,C] = 1 + \sum_{u\in C^\circ} P_u \\ = 1 + \sum_{u\in C^\circ} \prod_{v\in \mathrm{path}(u)\setminus\{o\}} p_\kappa(x_v \mid h_v).\end{aligned} \tag{1}$$

Here $G = 1 + L$ includes the root contribution and p_κ is the target distribution after decoding transform κ. Complete cycle time includes drafting, executed decisions, and verification:

$$T = T^{\mathrm{d}} + T^{\pi} + T^{\mathrm{v}}, \qquad \widehat{\eta}(S,\omega,C) = \frac{\bar{G}(S,\omega,C)}{\widehat{T}(S,\omega,C)}. \tag{2}$$

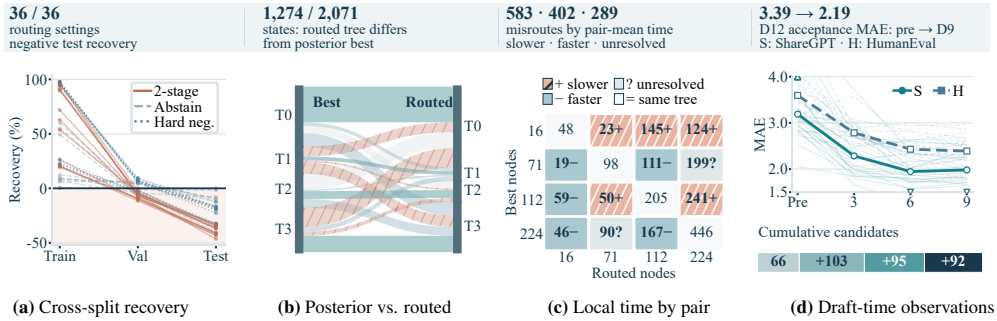
Verification time includes pruning, input collation, target evaluation, commitment, and cache updates. The device-calibrated latency prediction model $\widehat{T}$ predicts complete cycle time from context, topology, and execution layout; its specification is in Appendix A.1.

3.2 From Posterior Structural Gains to Drafting-Time Decisions

Posterior verification records contain target-side acceptance probabilities obtained by offline verification of recorded draft candidates. Posterior optimization uses these records to estimate throughput and select the best reusable tree topologies in the supported candidate domain. Sibling-rank addresses align shapes across states, and their supported union forms a supertree $\mathcal{U}$ with legal subtrees $\mathcal{H}$. The historical solver partitions structures by latency-determining features; each group then has a fixed predicted-time denominator and additive node progress. Solving the legal-subtree integer program within every group and comparing their optima gives the best single shared tree in this finite domain. A portfolio assigns each recorded history its best supported member:

$$\max_{\mathcal{P}\subseteq\mathcal{H},\ 1\le|\mathcal{P}|\le K,\ \mathcal{P}\cap\mathcal{D}_i\neq\varnothing\ \forall i:w_i>0} \sum_i w_i \max_{C\in(\mathcal{P}\cap\mathcal{D}_i)} \widehat{\eta}_i(C), \tag{3}$$

Here w_i weights history (S_i,ω_i), $\mathcal{D}_i$ contains its supported trees, and $\widehat{\eta}_i$ is posterior surrogate throughput. The inner maximum selects each history's member using recorded outcomes; $K = 1$ gives one shared tree. Alternating history assignment and member optimization produced a four-tree



(a) Cross-split recovery **(b)** Posterior vs. routed **(c)** Local time by pair **(d)** Draft-time observations

Figure 1: Pre-drafting routing and drafting-time information. (a) Posterior-gain recovery across training, validation, and internal test splits for 36 router configurations. (b) Posterior-best versus routed assignments in a separate four-template study. (c) Assignment disagreements and the sign of each template pair's mean local draft-and-verify time difference. (d) Request-level prediction error for depth-12 accepted tokens as observations accumulate, with cumulative candidate counts below. Panel (d) uses a separate historical development collection; complete-request execution assesses online throughput.

portfolio whose surrogate throughput exceeded the fixed-tree reference by about 9.3% and 6.5% on two held-out state sets.

Pre-drafting routing selects a portfolio member before verification outcomes are known. Of 71 examined router configurations, the 36 with paired internal tests lost their training gains on those tests; the best held-out router recovered 2.0% of the posterior gain. Verification-informed selection recovered 50%–76% under the same switch gates and actions, highlighting the information bottleneck in pre-drafting routing (Figure 1; protocols and positive external proxy cases in Appendix D).

Drafting provides new candidate scores and a generated prefix reusable by later actions. In historical development requests, intermediate observations reduced eventual accepted-count prediction error from 3.39 to 2.19 tokens; a separate fixed-action probe increased local surrogate throughput from 149.57 to 151.81 tokens/s. Mean additional accepted progress fell from 0.96 to 0.04 tokens between early and late decision points. As in LTD (Zhang et al., 2026a), decisions during drafting can exploit new information, with its value weighed against the incremental cost of acquiring and using it.

Structure determines the verification choices available at each decision point. On one staged draft structure, adaptive stopping and a fixed depth-9 exit achieved nearly equal long-generation throughput: 133.756 and 133.681 tokens/s. This result motivates optimizing verification choices alongside stopping decisions, accounting for both information and complete execution cost.

3.3 Joint Structure–Policy Optimization Objective

Draft structures determine observable information and feasible actions; online decisions determine their value. The ideal design therefore jointly chooses candidate generation, decision opportunities, and termination. Let $\mathfrak{G}$ be the executable configurations under specified models, verifiers, memory bounds, and computational constraints. A configuration $\Gamma \in \mathfrak{G}$ specifies generation and verification rules and execution layout; a policy $\pi \in \Pi_{\mathcal{F}}(\Gamma)$ chooses among actions available at decision step j using only current information $\mathcal{F}_j$. With jointly induced progress $G(\Gamma, \pi)$ and complete time $T(\Gamma, \pi)$, define their *reference efficiency* on a fixed population of cycle-start states as expected progress divided by expected time:

$$\eta^*_{\mathrm{ref}} = \sup_{\Gamma \in \mathfrak{G},\, \pi \in \Pi_{\mathcal{F}}(\Gamma)} \frac{\mathbb{E}_{\mathrm{ref},\omega,\pi,\zeta}[G(\Gamma,\pi)]}{\mathbb{E}_{\mathrm{ref},\omega,\pi,\zeta}[T(\Gamma,\pi)]}. \tag{4}$$

The reference population is generated by a frozen rollout generator under a fixed request distribution; ζ includes verification and device-time randomness, and feasible expected time is positive. Equation (3) averages historywise best surrogate throughput values chosen with posterior knowledge. Equation (4) instead evaluates decisions made with the information then available and divides expected progress by expected wall-clock time. Complete-request throughput subsequently tests efficiency on trajectories induced by each deployed configuration.

4 Methodology

The ideal objective couples draft structure, online information, and decision positions. OGSPO first solves the posterior position-selection problem under complete execution cost (**U0**). With decision positions fixed, posterior-based action net-gain estimates guide policy optimization on a fixed draft structure (**U1**), while the learned policy's reach and stop probabilities determine the

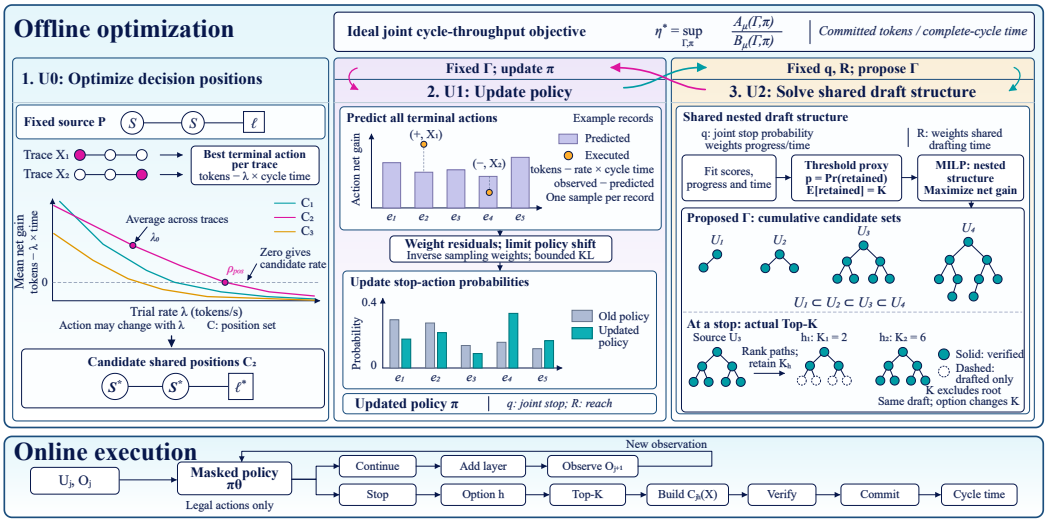


Figure 2: OGSPO overview. Offline optimization selects decision positions and alternates policy and shared draft structure optimization. Policy updates use posterior action net-gain estimates and residual-corrected policy comparisons; structural updates use reach/stop probabilities. Online execution continues drafting or stops and verifies a subtree selected by a verification-size option.

structural objective (**U2**). Both updates use a common progress–time criterion; each new structure is returned to policy optimization before independent comparison of the resulting pair (Figure 2).

4.1 Throughput Reformulation and Conditional Optimization

Let $\mathcal{U}$ be the candidate supertree, and identify each node u by its sibling-rank path (address). A shared nested draft structure records the cumulative set of generated paths at each decision point: $\Gamma = (U_1, \ldots, U_J)$ with $U_1 \subseteq \cdots \subseteq U_J$. The finite feasible set $\mathcal{G}_{\text{exe}}$ contains structures satisfying the execution constraints in Appendix A.3, and each structure induces a complete configuration $\phi(\Gamma) \in \mathfrak{G}$. The implementable policy class $\Pi(\Gamma)$ uses only observations and actions available at the selected positions, and is therefore a subset of the ideal class $\Pi_{\mathcal{F}}(\phi(\Gamma))$. ν_μ denotes the full-trace distribution induced by the reference cycle-start population μ. For a structure–policy pair (Γ, π), write A_μ for expected committed tokens and $B_\mu > 0$ for expected complete-cycle time, so its throughput is $\eta_\mu = A_\mu / B_\mu$.

Within each update block, we freeze a reference rate ρ. The progress–time net gain

$$J_\rho(\Gamma, \pi) = A_\mu(\Gamma, \pi) - \rho B_\mu(\Gamma, \pi), \qquad \eta_\mu(\Gamma, \pi) - \rho = \frac{J_\rho(\Gamma, \pi)}{B_\mu(\Gamma, \pi)} \tag{5}$$

is positive exactly when the pair exceeds the reference rate. From the current structure Γ_0, the conditional targets are

$$\max_{\pi \in \Pi(\Gamma_0)} J_{\rho_{\text{pol}}}(\Gamma_0, \pi), \qquad \max_{\Gamma \in \mathcal{G}_{\text{exe}}} \widetilde{J}_{\rho_{\text{str}}}(\Gamma; q^0). \tag{6}$$

Here ρ_{pol} is the rate before policy learning. On Γ_0, the updated policy π^0 induces two quantities (Section 4.3): q^0_{jh}, the joint probability of reaching decision point j and stopping with verification-size option h, and R^0_j, the probability of reaching decision point j. These state-conditioned reach and stop-action probabilities are held fixed during structural optimization. The structural rate is the updated pair's throughput, $\rho_{\text{str}} = \eta_\mu(\Gamma_0, \pi^0)$. The resulting structure–policy pair is independently compared after policy adaptation (Section 4.4).

4.2 U0: Posterior Optimization of Decision Positions

On a fixed source draft structure, full traces reveal each history's terminal-action values. Enabling a decision point exposes actions to all histories and charges those reaching it. We select shared decision positions, including terminal depth, by the enabled actions' posterior progress and complete time (Figure 2).

Let $P = (V_d)_{d \in \mathcal{D}}$ be a fixed source draft structure whose cumulative candidate sets cover all executable depths in $\mathcal{D}$. A choice $C = (\ell, S)$ specifies both a terminal depth $\ell \in \mathcal{D}$ and optional decision points $S \subseteq \{d \in \mathcal{D} : d < \ell\}$. At an optional decision point, the controller continues drafting

or stops with a verification-size option h; at depth d, this verification-size option retains $K_h(n_d)$ of the n_d generated non-root candidates in a legal subtree. Continuing reuses the generated prefix, and the terminal depth forces a stop. Output allowances and EOS mask unavailable actions.

From each cycle start in μ, we record a full draft trace X along P. For every legal terminal action $e = (d, h)$ under C, target-side probabilities and the verifier's actual rule give its conditionally expected committed progress $\bar{G}_X(e)$. The complete-cycle latency prediction model $\widehat{T}_X(C, e) > 0$ charges drafting through d, each reached decision point's stop or continuation cost, and terminal pruning, verification, commitment, and cache operations. Drafting cost is calibrated between successive active depths, so changing C can alter the time of a terminal action even when its verification tree is unchanged. At deployment, the controller observes only O_d: the confirmed prefix, candidates and draft scores produced by depth d, remaining output allowance, and decoding settings.

Let $\mathcal{C}(P)$ contain all executable choices of decision positions on P, with no preset terminal depth or decision point count, and let $\mathcal{E}_C(X)$ contain the legal terminal actions of C on trace X. The unrestricted posterior decision-position optimization problem has root

$$\Phi_P(\lambda) = \max_{C \in \mathcal{C}(P)} \mathbb{E}_{X \sim \nu_\mu} \left[\max_{e \in \mathcal{E}_C(X)} \left\{ \bar{G}_X(e) - \lambda \widehat{T}_X(C, e) \right\} \right], \qquad \Phi_P(\rho_{\text{pos}}) = 0. \tag{7}$$

With positive complete time, ρ_{pos} is the largest posterior ratio of expected progress to expected time on the fixed source structure. The inner maximum selects each trace's terminal action; the outer maximum selects shared depths, balancing their collective action value against availability cost.

For the finite solve, we retain the terminal-only choice and optimize two to six active positions. The upper count is a cost-informed budget: as enabled positions accumulate, controller calls and draft-segment boundaries can consume as much time as the drafting work that further decisions might save. Call this family $\mathcal{C}_{\leq 6}(P)$. Binary variables choose its shared positions and each trace assigns a legal terminal action. Calibrated segment and decision costs are charged only when that trace reaches them, producing a mixed-integer linear subproblem at each λ and a bound on the budgeted posterior optimum (Appendix A.1).

For fixed positions, posterior-optimal throughput bounds online policy throughput under the same source structure and complete-cost model. Controllers learn to identify valuable terminal actions from current observations. Independent online comparisons on the same reference population select the deployed positions from candidate sets, defining the policy update's legal action set.

4.3 U1: Policy Optimization with Residual Correction

Decision point j lies at depth d_j. With Γ fixed, offline record X_i supports evaluation of every legal terminal action $e = (j, h)$: target probabilities estimate progress $\bar{G}_{i,e}$, and the latency prediction model estimates complete-cycle time $\widehat{T}_{i,e}$. Evaluating the policy at successive pre-decision observations gives $q_{\theta,e}(X_i)$, the probability of reaching j and stopping through h.

At decision point j, let $a_j(O_j)$ be the continuation probability and $\beta_{jh}(O_j)$ the verification-size option probability conditional on stopping. Then

$$R_j(X) = \prod_{\ell < j} a_\ell(O_\ell), \qquad q_{jh}(X) = R_j(X)[1 - a_j(O_j)]\beta_{jh}(O_j), \quad R_1 = 1,\ a_J = 0. \tag{8}$$

We estimate the net gain of every legal terminal action and correct policy-improvement estimates using the residual from one sampled execution per selected record (Figure 2). For policies (θ, θ_0), let $d_{i,e} = q_{\theta,e}(X_i) - q_{\theta_0,e}(X_i)$ and $\widehat{z}_{i,e} = \bar{G}_{i,e} - \rho_{\text{pol}} \widehat{T}_{i,e}$. Records selected for execution ($i \in \mathcal{I}$) have reference weights w_i and known inclusion probabilities s_i. On each selected record, we execute one action E_i with known behavior probability $b_{i,E_i} > 0$ and observe net gain $Y_i = G_i - \rho_{\text{pol}} T_i$. The execution conditions on the exact recorded trace X_i, including its candidates and observations. The estimated policy improvement is

$$\widehat{\Delta}_{\rho_{\text{pol}}}(\theta; \theta_0) = \sum_i w_i \sum_e d_{i,e} \widehat{z}_{i,e} + \sum_{i \in \mathcal{I}} \frac{w_i}{s_i} \frac{d_{i,E_i}}{b_{i,E_i}} \left(Y_i - \widehat{z}_{i,E_i} \right). \tag{9}$$

The inverse-probability-weighted execution residual corrects the full-action estimate of policy improvement. Conditional unbiasedness holds when the policy pair is fixed before residual collection, execution is sampled on the same reference traces with known probabilities, and every action whose probability changes has positive sampling support. Its expectation then equals the true difference in net gain even when the model values are biased (Appendix A.2). We maximize it subject to an average Kullback–Leibler (KL) trust-region constraint, $\mathbb{E}_{X \sim \nu_\mu} D_{\text{KL}}(q_\theta(\cdot \mid X) \| q_{\theta_0}(\cdot \mid X)) \leq \delta_\pi$. The update acts directly on terminal-action probabilities, without a separately learned value critic.

4.4 U2: Shared Draft Structure Optimization

The updated reach and stop-action probabilities on Γ_0 weight progress and cost as in Eq. (8) (Figure 2). Let $w_{jh} = \mathbb{E}_{\nu_\mu} q^0_{jh}(X)$ be the mass of terminal action (j, h). For $w_{jh} > 0$, terminal progress and time are averaged under $\nu_{jh}(dX) = q^0_{jh}(X)\nu_\mu(dX)/w_{jh}$, whereas stage cost is averaged under ν_μ with weight $R^0_j(X)$.

For action (j, h), let $x_{ju} = \mathbf{1}\{u \in U_j\}$ indicate address membership at decision point j, with $n_j = \sum_{u \neq o} x_{ju}$. Hard top-$K$ pruning couples nodes within a record, so we relax its per-record budget to an expected budget under a common score threshold t_{jh}. Under the action-use distribution, let Q_{jhu} be the draft path score, g_{jhu} its target-side progress contribution when retained, $p_{jhu} = \Pr(Q_{jhu} > t_{jh}) + \tau_{jh} \Pr(Q_{jhu} = t_{jh})$, and $\bar{g}_{jhu} = \mathbb{E} g_{jhu}$. A common tie fraction $\tau_{jh} \in [0, 1]$ meets the expected budget at threshold atoms (Appendix A.3). The proxy for committed progress is

$$\sum_{u \neq o} x_{ju} p_{jhu} = K_h(n_j), \qquad y_{jhu} = x_{ju} p_{jhu}, \qquad \widetilde{A}_{jh} = 1 + \sum_{u \neq o} \bar{g}_{jhu} y_{jhu}. \tag{10}$$

Here y_{jhu} is probability-weighted node inclusion; $\widetilde{A}_{jh}$ includes the root contribution. The proxy approximates the joint expected contribution under hard top-K by the product of marginals $\bar{g}_{jhu} p_{jhu}$. The approximation gap has two sources: threshold inclusion can differ from statewise top-K selection, and inclusion can covary with target-side progress. Appendix A.3 bounds the first source and estimates both on paired traces from the same starts.

Let $\widehat{v}_{jh}(\Gamma)$ be predicted terminal time under ν_{jh}, and $\widehat{s}_j(X, \Gamma)$ the incremental generation and control time at j. Holding the updated reach and stop-action probabilities q^0, R^0 fixed, we optimize

$$\widetilde{J}_{\rho_{\mathrm{str}}}(\Gamma; q^0) = \sum_{j,h: w_{jh} > 0} w_{jh} \left[\widetilde{A}_{jh} - \rho_{\mathrm{str}} \widehat{v}_{jh}(\Gamma) \right] - \rho_{\mathrm{str}} \sum_j \mathbb{E}_{\nu_\mu} \left[R^0_j(X) \widehat{s}_j(X, \Gamma) \right]. \tag{11}$$

Every enabled action must remain legal, including actions with zero probability under q^0. A node's value therefore depends on the nodes it may displace at the pruning boundary (Appendix A.3). The solver optimality-gap bound covers this fitted structural proxy objective, and candidate acceptance uses actual top-K pruning.

After each structural update, action net-gain estimates are recomputed and U1 optimizes the policy on the updated draft structure. An independent gain comparison decides whether the updated pair becomes the next baseline; iteration ends on a gain bound, stabilization, or budget exhaustion.

5 Experiments

5.1 Setup

We evaluate EAGLE-3 target–draft pairs based on Llama-3.1-8B (Grattafiori et al., 2024), Vicuna-13B (Chiang et al., 2023), and Qwen3-32B (Qwen Team, 2025). All three run on H800; Llama-3.1-8B also runs on Spark GB10, RTX 5000, and Ascend 910B. The evaluated configurations cover temperatures zero and one. The four common workloads are EAGLE QA (SafeAILab, n.d.), GSM8K (Cobbe et al., 2021), Alpaca (Taori et al., 2023), and MT-Bench (Zheng et al., 2023). We compare native EAGLE-3 (Li et al., 2025), its greedy-search tree variant (GS), Learning to Draft (LTD) (Zhang et al., 2026a), and **OGSPO** on matched requests at batch size one. Each model–device–temperature setting uses a separately fitted configuration. Our primary metric is task-level mean throughput in committed tokens per second. The four-workload summary averages the task means equally. Speedup over autoregressive decoding, accepted length, and cycle time separate progress from cost. The shaded rows in Table 1 give posterior references for decision selection and statewise tree selection.

5.2 Main results

Table 1 reports two complete Llama-8B/H800 and four compact cross-setting comparisons; Appendix E gives all twelve settings. Across all twelve settings, OGSPO improves mean throughput over native EAGLE-3 by 4.8–19.3%. Its gain over LTD is 2.0–8.9% in the ten settings with a matched LTD comparison. On Llama-8B/H800 at temperature zero, OGSPO reaches 145.4 tokens/s, compared with 131.4 for EAGLE-3, 141.1 for GS, and 138.3 for LTD. Lower cycle cost supplies a second route to higher throughput. On QA and GSM8K in that setting, OGSPO accepts fewer tokens per cycle than LTD but raises throughput from 115.6 to 119.8 and from 142.3 to 153.3 tokens/s, respectively. Cycle times fall from 54 to 50 ms and from 51 to 44 ms. Across H800, OGSPO exceeds GS in all 24 matched task comparisons and LTD in 15 of 16; the remaining comparison is Vicuna-13B on

Table 1: Throughput (TPS, tokens/s), speedup over autoregressive decoding (Spd.), and accepted length / cycle time (AL/T, ms) on three tasks and the four-task mean. The first two settings include shaded posterior decision and statewise tree references. The full twelve-setting matrix is in Appendix E.

Setting	Method	QA			GSM8K			MT-Bench			Four-task mean		
		TPS	Spd.	AL/T	TPS	Spd.	AL/T	TPS	Spd.	AL/T	TPS	Spd.	AL/T
Llama-8B H800 / $T = 0$	EAGLE-3	110.1	3.13	5.6/53	137.0	3.86	6.4/48	135.3	3.83	6.4/49	131.4	3.72	6.3/50
	GS	116.7	3.32	5.9/52	148.4	4.19	7.0/48	145.6	4.12	6.9/49	141.1	4.00	6.8/50
	LTD	115.6	3.29	6.1/54	142.3	4.01	7.1/51	143.4	4.05	7.3/53	138.3	3.92	7.1/53
	OGSPO	**119.8**	**3.41**	**5.7/50**	**153.3**	**4.32**	**6.6/44**	**149.1**	**4.22**	**6.7/46**	**145.4**	**4.12**	**6.5/47**
	Post. dec.	126.2	3.59	5.9/48	160.0	4.51	6.8/44	155.1	4.38	6.9/46	152.4	4.32	6.8/46
	Oracle	141.8	4.03	6.3/46	171.4	4.83	7.4/45	173.9	4.92	7.5/44	168.7	4.78	7.3/45
Llama-8B H800 / $T = 1$	EAGLE-3	80.9	3.09	5.1/65	117.6	3.97	5.9/52	95.2	3.89	5.7/62	102.0	3.74	5.7/59
	GS	83.1	3.18	5.4/67	112.8	3.81	6.4/58	92.9	3.80	6.1/68	101.9	3.74	6.2/63
	LTD	81.0	3.10	5.5/71	115.8	3.91	6.6/58	93.6	3.83	6.5/71	101.8	3.74	6.4/65
	OGSPO	**89.4**	**3.42**	**5.4/62**	**127.7**	**4.31**	**5.8/47**	**101.3**	**4.14**	**5.7/58**	**110.8**	**4.07**	**5.9/56**
	Post. dec.	96.4	3.69	5.3/57	137.0	4.62	5.9/45	111.5	4.56	5.7/53	119.9	4.40	5.9/51
	Oracle	113.7	4.35	5.2/47	152.7	5.15	6.1/41	144.0	5.89	5.8/42	141.4	5.19	5.9/43
Vicuna-13B H800 / $T = 1$	EAGLE-3	91.7	3.14	4.6/53	104.8	4.12	6.1/60	109.4	4.19	6.2/58	101.4	3.80	5.7/58
	GS	93.1	3.19	5.5/61	108.2	4.25	7.4/70	110.2	4.22	7.6/71	103.5	3.88	6.9/68
	LTD	94.4	3.24	5.3/58	109.2	4.29	7.1/66	118.2	4.52	7.7/67	107.3	4.02	6.7/64
	OGSPO	**95.8**	**3.28**	**4.6/50**	**118.4**	**4.65**	**5.7/50**	**117.6**	**4.50**	**5.9/52**	**109.8**	**4.12**	**5.5/52**
Qwen-32B H800 / $T = 0$	EAGLE-3	28.2	1.71	2.3/88	40.9	2.54	3.4/88	35.5	2.21	2.7/81	33.1	2.06	2.7/88
	GS	30.2	1.83	2.3/84	47.0	2.93	3.5/80	37.9	2.36	2.7/78	37.2	2.31	2.8/80
	OGSPO	**32.4**	**1.97**	**2.5/85**	**50.1**	**3.12**	**3.5/75**	**40.3**	**2.51**	**2.7/73**	**39.5**	**2.45**	**2.9/79**
Llama-8B Spark / $T = 0$	EAGLE-3	38.8	3.18	5.4/126	48.5	3.82	6.2/126	47.2	3.73	6.4/130	46.3	3.73	6.2/127
	LTD	40.3	3.31	6.1/155	49.6	3.91	7.1/148	49.0	3.88	7.3/154	48.1	3.87	7.1/151
	OGSPO	**41.0**	**3.36**	**5.4/121**	**52.7**	**4.15**	**5.9/109**	**50.5**	**4.00**	**6.1/116**	**49.8**	**4.01**	**6.1/117**
Llama-8B 910B / $T = 0$	EAGLE-3	77.0	3.21	5.6/75	101.4	3.77	6.4/66	97.4	3.78	6.4/68	93.3	3.73	6.3/70
	LTD	80.0	3.34	6.1/78	103.6	3.85	7.1/71	101.2	3.93	7.3/74	96.8	3.87	7.1/75
	OGSPO	**83.7**	**3.49**	**5.9/73**	**112.0**	**4.17**	**6.4/58**	**105.9**	**4.11**	**6.5/63**	**102.6**	**4.10**	**6.6/66**

MT-Bench at temperature one. Beyond H800, four-workload mean throughput rises from 46.3 to 49.8 tokens/s on Spark GB10 and from 93.3 to 102.6 on Ascend 910B, comparing EAGLE-3 with OGSPO. In the complete Llama-8B/H800 comparison, posterior decision selection reaches 152.4 tokens/s, and statewise tree selection reaches 168.7.

5.3 Analytical results

Policy adaptation increases the throughput gained from a new tree. Across six paired groups, replacing the original tree while freezing the policy yields a median throughput ratio of 1.022 relative to the original pair (Figure 3a). Adapting the policy on the new tree raises this ratio to 1.043, whereas equal additional training on the original tree reaches 1.016. One group slows under the frozen policy, but all six improve after adaptation. Cross-evaluation in (b) shows that the preferred tree varies across policy snapshots and the preferred policy varies across trees. The finite-domain search in (c) raises the feasible value and tightens its certified bound; on a separate candidate set, the mean-plus-variance proxy in (d) tracks evaluated gains more closely than the mean-only proxy.

Residual-corrected feedback reaches demanding efficiency targets with less offline work. At an offline budget of 12 normalized units, 10 of 16 residual-correction runs reach relative efficiency 1.08, compared with three sampled-feedback runs and no model-only runs (Figure 4a,b). At budget 15, the counts rise to 16, nine, and one; all three methods reach the lower 1.05 target in every run by budget 12. Residual-corrected updates also extend into the high signal-to-noise, high gradient-agreement region in (c), while (d) shows their larger later gain per unit of offline budget through most of training.

Request-level gains accompany both early and late drafting exits. Mean LTD-to-OGSPO time ratios exceed one in the occupied input–output length bins on QA and HumanEval (Chen et al., 2021) (Figure 5a,b). Across 80 matched requests per task, the median ratio is 1.082 on QA and 1.119 on HumanEval. Of the drafting episodes still active at each drafting step, 28.6% exit at the first step; the conditional exit probability remains between 8.6% and 12.6% at steps 13–16 (c).

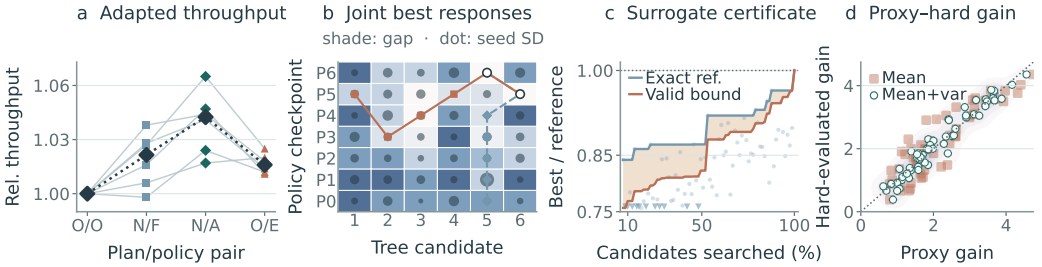
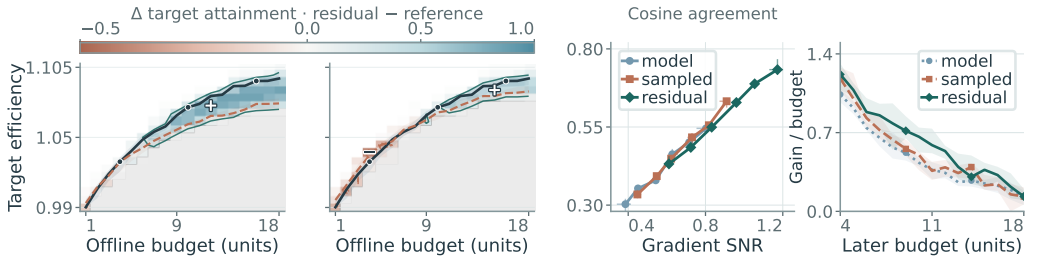


Figure 3: Structure–policy adaptation. (a) Throughput relative to the original structure–policy pair (O/O): new/frozen (N/F), new/adapted (N/A), and original/extra-trained (O/E). (b) Dashed/solid paths identify each policy's best tree/each tree's best policy; rings mark mutual best responses, shading their summed unilateral gap, and dot size seed variation. (c) Best feasible value normalized by the finite-domain optimum (blue) or a valid upper bound (orange); triangles mark off-range candidates. (d) Proxy versus hard-evaluated gain on separate candidates.



(a) vs. model only **(b)** vs. sampled **(c)** Gradient quality **(d)** Marginal return

Figure 4: Feedback and offline learning cost. Efficiency is throughput relative to a shared reference; budget is normalized. (a,b) Curves show 50%-attainment frontiers; color shows attainment-probability differences (residual correction minus reference) across 16 matched seeds. (c) Gradient signal-to-noise ratio versus half-batch cosine agreement. (d) Later efficiency gain per budget unit.

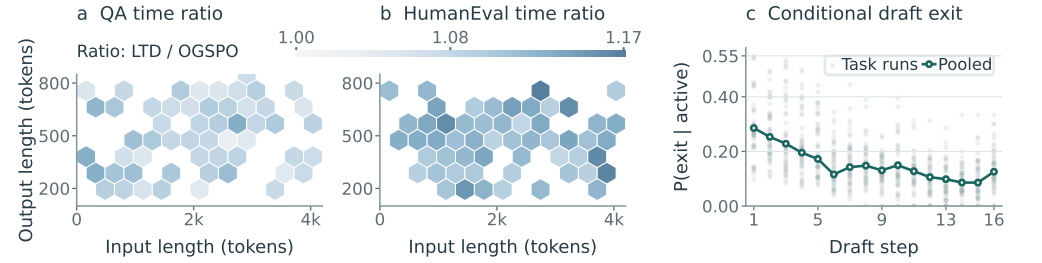


Figure 5: Request time and drafting exits. (a,b) Mean LTD/OGSPO time ratio per occupied input–output length bin, with a shared color scale for QA and HumanEval. (c) Exit probability conditional on still drafting; the line pools task runs and pale points denote individual runs.

6 Conclusion

We investigated how drafting-time decisions realize posterior structural gains in tree-based speculative decoding. Preliminary experiments identify the difficulty of pre-drafting template selection and the value of richer observations and shared computation during drafting. OGSPO jointly optimizes draft structures and online policies under a common throughput objective. It selects decision depths from posterior verification records and modeled costs. Policy learning combines surrogate net-gain estimates with same-trace execution feedback that corrects policy-improvement estimates. Reach/stop probabilities then weight mixed-integer optimization of the shared draft structure, followed by revaluation and policy adaptation. Our analysis characterizes information value, establishes conditional unbiasedness of policy-improvement estimates under the sampling protocol, and bounds the structural proxy's finite-domain optimality gap under fixed reach/stop probabilities.

Across twelve configurations, OGSPO improves four-workload mean complete-request throughput over EAGLE-3 by 9.2% on average and up to 19.3%. Controlled analyses show that policy adaptation increases structural gains and residual-corrected feedback reaches higher efficiency targets with less offline work. OGSPO connects posterior structural gains to online efficiency through joint structure–policy optimization as information emerges during drafting.

AI USE STATEMENT

Generative AI tools assisted with translation, manuscript drafting and revision, literature comparison, terminology consistency, reference formatting, and LaTeX and figure-layout preparation. During author-led research discussions, they also supported refinement of the problem formulation and methodology, examination and presentation of mathematical claims and proof arguments, and review of experimental design and analysis.

The authors considered these suggestions through iterative discussion and revision, determined the research direction and methodological choices, and selected the content incorporated into the manuscript. The authors take responsibility for the final text, claims, results, and artifacts.

REFERENCES

Sudhanshu Agrawal, Wonseok Jeon, and Mingu Lee. AdaEDL: Early draft stopping for speculative decoding of large language models via an entropy-based lower bound on token acceptance probability. In *Proceedings of the 4th NeurIPS Efficient Natural Language and Speech Processing Workshop*, volume 262 of *Proceedings of Machine Learning Research*, pp. 355–369. PMLR, 2024. URL https://proceedings.mlr.press/v262/agrawal24a.html.

Zachary Ankner, Rishab Parthasarathy, Aniruddha Nrusimha, Christopher Rinard, Jonathan Ragan-Kelley, and William Brandon. Hydra: Sequentially-dependent draft heads for medusa decoding. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=FbhjirzvJG.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling, 2023. URL https://arxiv.org/abs/2302.01318.

Mark Chen et al. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Zhuoming Chen, Avner May, Ruslan Svirschevski, Yuhsun Huang, Max Ryabinin, Zhihao Jia, and Beidi Chen. Sequoia: Scalable and robust speculative decoding. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4116. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ea1f5f0878d43ff4fb8bf64ef4a2326c-Abstract-Conference.html.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing GPT-4 with 90%* ChatGPT quality. LMSYS Org., March 2023. URL https://www.lmsys.org/blog/2023-03-30-vicuna/.

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. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Werner Dinkelbach. On nonlinear fractional programming. *Management Science*, 13(7):492–498, 1967. doi: 10.1287/mnsc.13.7.492. URL https://doi.org/10.1287/mnsc.13.7.492.

Aaron Grattafiori et al. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Yue Guan, Changming Yu, Shihan Fang, Weiming Hu, Zaifeng Pan, Zheng Wang, Zihan Liu, Yangjie Zhou, Yufei Ding, Minyi Guo, and Jingwen Leng. Yggdrasil: Bridging dynamic speculation and static runtime for latency-optimal tree-based LLM decoding. In *Advances in Neural Information Processing Systems 38*, 2025. doi: 10.52202/085713-2901. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/7d66be507114ffe38c3d3eb528454952-Abstract-Conference.html.

Shijing Hu, Jingyang Li, Zhihui Lu, and Pan Zhou. Bridging draft policy misalignment: Group tree optimization for speculative decoding. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/b6e67ae290635d0874c4cb43ba2a2cfb-Abstract-Conference.html.

Kaixuan Huang, Xudong Guo, and Mengdi Wang. Specdec++: Boosting speculative decoding via adaptive candidate lengths. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Y131N9fUbU.

Sungkyun Kim, Jaemin Kim, Dogyung Yoon, Jiho Shin, Junyeol Lee, and Jiwon Seo. Speculative verification: Exploiting information gain for speculative decoding. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 36290–36307. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1809. URL https://aclanthology.org/2026.findings-acl.1809/.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Tianyi Li, Yaxin Luo, Xinyi Shang, and Zhiqiang Shen. DARTree: Speculative diffusion decoding with autoregressive draft trees, 2026. URL https://arxiv.org/abs/2608.13524.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-2: Faster inference of language models with dynamic draft trees. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7421–7432. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.422. URL https://aclanthology.org/2024.emnlp-main.422/.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-3: Scaling up inference acceleration of large language models via training-time test. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025. doi: 10.52202/085713-4562. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/c7b5a35ea98b62512a869c19ea7b03cb-Abstract-Conference.html.

Saw S. Lin and Jyh-Shing Roger Jang. Dominotree: Conditional tree-structured drafting with domino for speculative decoding, 2026. URL https://arxiv.org/abs/2607.08642v3.

Hongyi Liu, Jiaji Huang, Zhen Jia, Youngsuk Park, and Yu-Xiang Wang. Not-a-Bandit: Provably no-regret drafter selection in speculative decoding for LLMs. In *International Conference on Learning Representations*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/20563b8508ba42e1b688d922e926ee26-Abstract-Conference.html.

Siran Liu, Yang Ye, Qianchao Zhu, Zane Cao, and Yongchao He. HeteroSpec: Leveraging contextual heterogeneity for efficient speculative decoding. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12930–12947, San Diego, California, 2026b. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.589. URL https://aclanthology.org/2026.acl-long.589/.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Shi, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. SpecInfer: Accelerating large language model serving with tree-based speculative inference and verification. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, pp. 932–949. Association for Computing Machinery, 2024. doi: 10.1145/3620666.3651335. URL https://dl.acm.org/doi/10.1145/3620666.3651335.

Lehan Pan, Ziyang Tao, Xiao Wang, Zhimeng Zhao, Meng Wang, and Yanyong Zhang. Making every verified token count: Adaptive verification for moe speculative decoding, 2026. URL https://arxiv.org/abs/2605.00342.

Qwen Team. Qwen3-32B model card. Official Qwen model repository, 2025. URL https://huggingface.co/Qwen/Qwen3-32B. Accessed 2026-09-26.

SafeAILab. EAGLE qa evaluation questions. Official EAGLE repository dataset file, n.d. URL https://github.com/SafeAILab/EAGLE/blob/main/eagle/data/qa/question.jsonl. 80 items, question IDs 321–400; accessed 2026-09-25.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpaca: A strong, replicable instruction-following model. Stanford Center for Research on Foundation Models, March 2023. URL https://crfm.stanford.edu/2023/03/13/alpaca.html.

Rahul Thomas, Teo Kitanovski, Micah Goldblum, and Arka Pal. Dynamic delayed tree expansion for improved multi-path speculative decoding, 2026. URL https://arxiv.org/abs/2602.16994.

Jikai Wang, Yi Su, Juntao Li, Qingrong Xia, Zi Ye, Xinyu Duan, Zhefeng Wang, and Min Zhang. OPT-Tree: Speculative decoding with adaptive draft tree structure. *Transactions of the Association for Computational Linguistics*, 13:188–199, 2025. doi: 10.1162/tacl_a_00735. URL https://aclanthology.org/2025.tacl-1.8/.

Jingyi Wang, Jiaqi Huang, Zunnan Xu, Jun Zhou, Kehong Yuan, and Xiang Qian. Training-free adaptive speculative decoding via linguistic priors. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 21184–21194. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1065. URL https://aclanthology.org/2026.findings-acl.1065/.

Lifu Wang and Pan Zhou. SMART: When is it actually worth expanding a speculative tree? In *Computer Vision – ECCV 2026*, volume 17029 of *Lecture Notes in Computer Science*, pp. 228–244. Springer, 2026. doi: 10.1007/978-3-032-37627-5_13. URL https://link.springer.com/chapter/10.1007/978-3-032-37627-5_13.

Jiebin Zhang, Zhenghan Yu, Liang Wang, Nan Yang, Eugene Yu, Zheng Li, Yifan Song, Dawei Zhu, Xingxing Zhang, Furu Wei, and Sujian Li. Learning To Draft: Adaptive speculative decoding with reinforcement learning. In *International Conference on Learning Representations*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/34345e243156da67605d4b63d71c8d98-Abstract-Conference.html.

Shuai Zhang, Huachuan Qiu, Hongliang He, and Yong Dai. Cost-aware diffusion draft trees for speculative decoding, 2026b. URL https://arxiv.org/abs/2606.01813.

Situo Zhang, Hankun Wang, Da Ma, Zichen Zhu, Lu Chen, Kunyao Lan, and Kai Yu. AdaEAGLE: Optimizing speculative decoding via explicit modeling of adaptive draft structures, 2024. URL https://arxiv.org/abs/2412.18910. Preprint; accessed 2026-09-25.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623, 2023. doi: 10.52202/075280-2020. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

A METHOD DETAILS

This appendix derives the decision-position objective (U0), the residual-corrected policy update (U1), and the shared draft structure objective (U2). All three use the reference population in Equation (4), denoted by μ in Section 4.

A.1 DECISION-POSITION OBJECTIVE AND STAGE COSTS

For a choice of decision positions $C = (\ell, S)$, let $N(C, \psi)$ and $B(C, \psi) > 0$ be expected committed progress and complete time when a posterior rule ψ selects a legal terminal action on each full trace. For a fixed price λ, terminal-action choices are separable across traces, so $\max_\psi [N(C, \psi) - \lambda B(C, \psi)]$ equals the inner expectation in Eq. (7). Its sign is positive, zero, or negative according as the best ratio $N(C, \psi)/B(C, \psi)$ is above, equal to, or below λ. Maximizing over the full finite position choice family gives the unique root ρ_{pos}. Replacing that family by $\mathcal{C}_{\le 6}(P)$ gives the root for the cost-budgeted problem. This is the standard parametric transformation for a ratio objective (Dinkelbach, 1967).

For the empirical solve, use reference weights $\omega_i \ge 0$ with $\sum_i \omega_i = 1$. Its objective replaces the expectation in Eq. (7) by $\sum_i \omega_i \max_e \{\bar{G}_i(e) - \lambda \widehat{T}_i(C, e)\}$. The population and empirical roots each refer to their own reference distribution and calibrated cost model. Finiteness and a uniform bound $\widehat{B} \ge B_{\min} > 0$ give, for $\lambda_2 > \lambda_1$,

$$\Phi_P(\lambda_2) \le \Phi_P(\lambda_1) - (\lambda_2 - \lambda_1) B_{\min}.$$

Each feasible pair contributes an affine function of λ, so the finite empirical envelope is continuous and strictly decreasing. Evaluating it at the best feasible ratio gives zero, proving uniqueness of the root.

For a fixed active-position count $k \in \{1, \dots, 6\}$, let z_d select a depth and t_d mark the terminal depth. Both are binary and satisfy

$$\sum_d z_d = k, \quad \sum_d t_d = 1, \quad t_d \le z_d, \quad z_d \le \sum_{q \ge d} t_q.$$

Add a source vertex 0 preceding all executable depths. Binary arcs a_{pq} are allowed for $p < q$, including $p = 0$. The ordered path is encoded by

$$\sum_{p<d} a_{pd} = z_d, \qquad \sum_{q>d} a_{dq} = z_d - t_d, \qquad \sum_{q>0} a_{0q} = 1.$$

The depth order makes the path acyclic, and the flow equalities connect all selected depths to its unique terminal.

On trace i, split terminal-action weights into optional and terminal roles, ξ^{opt}_{idh} and $\xi^{\mathrm{term}}_{idh}$. Let $m^r_{idh} \in \{0, 1\}$ be the fixed legal-action mask for role r. Impose

$$\sum_{d,h,r} \xi^r_{idh} = 1, \quad 0 \le \xi^{\mathrm{opt}}_{idh} \le m^{\mathrm{opt}}_{idh}(z_d - t_d), \quad 0 \le \xi^{\mathrm{term}}_{idh} \le m^{\mathrm{term}}_{idh} t_d.$$

Feasible position choices provide a supported fallback on every positively weighted trace. Define $L_{iq} = \sum_{d \ge q,h,r} \xi^r_{idh}$ and $L^+_{iq} = \sum_{d>q,h,r} \xi^r_{idh}$. For calibrated segment cost $c_{i,pq}$, continuation-specific cost h^{cont}_{iq}, and stop-role cost v^r_{idh}, complete expected time on trace i is

$$B_i = \sum_{p<q} c_{i,pq} a_{pq} L_{iq} + \sum_q h^{\mathrm{cont}}_{iq} z_q L^+_{iq} + \sum_{d,h,r} v^r_{idh} \xi^r_{idh}.$$

Shared observation and control work is included in the segment ending at its decision point; role-specific work enters the second or third term. Every binary–continuous product $u = bL$ uses $0 \le u \le b$, $u \le L$, and $u \ge L - (1 - b)$. The fixed-price objective is $\sum_i \omega_i [\sum_{d,h,r} \bar{G}_i(d, h) \xi^r_{idh} - \lambda B_i]$. At fixed positions it is linear on each trace's terminal-action simplex, hence has a maximizing legal terminal action at a vertex. The resulting mixed-integer linear program equals the empirical posterior objective under this finite additive stage model. A valid scalarized solver upper bound δ at a feasible candidate's ratio bounds its remaining fitted ratio gap by $\delta / B_{\min}$. Scanning feasible counts includes the terminal-only case $k = 1$.

A price search maintains a bracket for the empirical root. A feasible positive net-gain incumbent establishes that the root exceeds the current price, while a valid nonpositive solver upper bound

establishes that it is at most that price. Prices with unresolved bound signs trigger further solution effort. A positive complete-time floor and a finite maximum progress supply an initial finite upper price. Bisection or the feasible ratio updates of the parametric method refine this bracket; the final scalarized gap provides the ratio certificate above.

Latency prediction model. The prediction model is calibrated separately for each device and execution layout. Its inputs describe context length, generated and verified node counts, tree depth and width, and packing layout. Complete time is the sum of nonoverlapping draft segments, reached decision operations, and terminal verification operations. If $f_a(X, C)$ is the feature vector of operation a, a finite calibration table or a fitted function $\widehat{\tau}_a(f_a)$ supplies its predicted cost. The resulting $\widehat{T}_X(C, e)$ follows the stage accounting below. Calibration targets measured operation time on the corresponding device; shared work is assigned once. A positive terminal-time floor ensures $\widehat{T}_X(C, e) \geq B_{\min} > 0$. For the integer programs, frozen calibration values are represented by finite lookup selectors or piecewise-linear constraints, so each feasible execution layout has a specified cost.

For a fixed arrival population and a common legal action set $\mathcal{A}_j$, let X denote the full draft record and $Z_a = \mathbb{E}[G_a - \rho T_a \mid X]$ the conditional expected remaining net gain of action a under its fixed downstream rule. This conditional expectation averages verification and device-time randomness; O_j is the observation available from X at decision point j. U0 evaluates this record-level reference using target probabilities and modeled costs. Costs paid before the decision point are shared by all actions. Define

$$V_j^{\text{fixed}} = \max_{a \in \mathcal{A}_j} \mathbb{E}_j Z_a, \qquad V_j^{\text{obs}} = \mathbb{E}_j \max_{a \in \mathcal{A}_j} \mathbb{E}_j[Z_a \mid O_j], \qquad V_j^{\text{post}} = \mathbb{E}_j \max_{a \in \mathcal{A}_j} Z_a. \tag{12}$$

Then $V_j^{\text{fixed}} \leq V_j^{\text{obs}} \leq V_j^{\text{post}}$. The first inequality follows because constant rules are included among observation-based rules; the second follows because observing each action outcome permits statewise selection. The middle term is the ideal value available from O_j; a restricted controller's realizable value is bounded by it and is measured by execution. When action masks vary by state, each comparison uses the corresponding common legal support.

For a finite common action set, write $m_a(O_j) = \mathbb{E}_j[Z_a \mid O_j]$ and select any unconditional maximizer a_* of $\mathbb{E}_j Z_a$. Then

$$\Delta_j^{\text{obs}} = \mathbb{E}_j\big[\max_a m_a(O_j) - m_{a_*}(O_j)\big].$$

The integrand is nonnegative. Consequently, zero information value is equivalent to the existence of one fixed action that is conditionally optimal almost surely. Positive value occurs when every fixed action is conditionally suboptimal on a set of positive probability. This formulation includes ties. In the binary case, putting $m = \mathbb{E}_j[M \mid O_j]$ gives strict positivity exactly when both $\Pr_j(m > 0) > 0$ and $\Pr_j(m < 0) > 0$.

For the binary continue-or-stop comparison, hold the downstream rule fixed after continuing, subtract the value of stopping, and set $M = Z_{\text{continue}} - Z_{\text{stop}}$. The fixed and observation-based values reduce to

$$V_j^{\text{fixed}} - \mathbb{E}_j Z_{\text{stop}} = (\mathbb{E}_j M)_+, \qquad V_j^{\text{obs}} - \mathbb{E}_j Z_{\text{stop}} = \mathbb{E}_j(\mathbb{E}_j[M \mid O_j])_+,$$

so the observation value is $\Delta_j^{\text{obs}} = V_j^{\text{obs}} - V_j^{\text{fixed}}$. For any continue rule $\gamma_j(O_j)$, the gap to hindsight selection is

$$M_+ - \gamma_j(O_j)M = [1 - \gamma_j(O_j)]M_+ + \gamma_j(O_j)(-M)_+. \tag{13}$$

The two terms are the losses from stopping when continuing is better and continuing when stopping is better. Costs shared by the actions, including the current decision call, cancel in or are excluded from M; costs incurred only after continuing remain in Z_{continue}. This local quantity describes the gap between posterior opportunities and decisions supported by current observations; Eq. (7) selects positions using the complete cost of every position choice.

The position procedure forms the feasible family from all executable depths of the frozen source draft structure. Each price-parametric solve jointly selects shared positions and trace-specific posterior terminal actions; its root gives the fitted posterior ratio for the selected position budget. Under the same source draft structure and fitted cost model, this budget-level optimum bounds every online controller using a choice of decision positions of that size. Independently evaluated controllers provide executable lower bounds for comparing candidate position choices.

For a fixed source draft structure and position choice C, every online controller is a feasible tracewise randomized terminal-action rule under the same action and cost model. Its net gain at the posterior

root is at most $\Phi_C(\rho_C^{\mathrm{post}}) = 0$, which implies $\eta_C(\pi) \le \rho_C^{\mathrm{post}}$. The posterior value therefore supplies an opportunity bound for training and comparing executable controllers. If valid simultaneous bounds satisfy $U_C^{\mathrm{post}} < L_{C'}^{\mathrm{online}}$, every controller on C is dominated by the evaluated controller on C'. The inference follows by chaining $\eta_C(\pi) \le \rho_C^{\mathrm{post}} \le U_C^{\mathrm{post}} < L_{C'}^{\mathrm{online}} \le \eta_{C'}(\pi')$. Each bound refers to the same source draft structure, reference distribution, and complete-cost model; bounds on physical execution incorporate the corresponding model uncertainty. Changing the source draft structure requires new values and position bounds. The selected position choice initializes the subsequent policy–structure updates.

All stage quantities below are conditional on the recorded trace X; averaging over reference traces is performed after weighting by its reach and stop-action probabilities. For a fixed position choice, let c_j be its nonoverlapping draft-segment cost, h_j the feature and control work shared by all actions at decision point j, h_j^{cont} the additional work specific to continuing, and v_{jh} the complete stop-specific cost for verification-size option h. The latter includes pruning, packing, verification, residual processing, commitment, cache update, and required next-root work. Complete action time is

$$T_{jh} = \sum_{\ell \le j} (c_\ell + h_\ell) + \sum_{\ell < j} h_\ell^{\mathrm{cont}} + v_{jh}.$$

With $R_{J+1} = 0$, Eq. (8) gives $\sum_h q_{jh} = R_j - R_{j+1}$ and $\sum_{\ell \ge j,h} q_{\ell h} = R_j$. Reordering the finite sums therefore yields

$$\sum_{j,h} q_{jh} T_{jh} = \sum_j R_j (c_j + h_j) + \sum_j R_{j+1} h_j^{\mathrm{cont}} + \sum_{j,h} q_{jh} v_{jh}. \tag{14}$$

Each segment and common control call is charged at its reach probability, each continuation-specific operation at the probability of continuing, and each terminal operation at its stop probability. Under the frozen reach and stop-action probabilities, $R_{j+1}^0 = R_j^0 a_j^0$, so the stage predictor $\widehat{s}_j = c_j + h_j + a_j^0 h_j^{\mathrm{cont}}$ gives exactly the reach/stop decomposition in Eq. (11), with v_{jh} containing stop-specific work. This mapping fixes the accounting convention for the structural proxy. For overlapping physical operations, calibration uses nonoverlapping increments or fits complete action time directly. A positive complete-time floor applies throughout the feasible domain.

A.2 Policy feedback estimators

Before collecting feedback, fix an offline full-draft trace X, two policies q_1, q_0, and all model values. The residual execution must be conditioned on this same trace by reproducing its recorded candidates and observations or by an equivalent conditional-sampling design. A same-start replay is used for paired configuration evaluation; same-trace residual collection additionally preserves the recorded candidates and observations. For each terminal action e, write $d_e(X) = q_1(e \mid X) - q_0(e \mid X)$ and let $b_e(X)$ be the known behavior probability, with $b_e(X) > 0$ whenever $d_e(X) \ne 0$. If action E is sampled from this behavior distribution and yields $Y = G - \rho T$, the paired correction is

$$\Psi(X, E, Y) = \sum_e d_e(X) \widehat{z}_e(X) + \frac{d_E(X)}{b_E(X)} [Y - \widehat{z}_E(X)]. \tag{15}$$

Writing $z_e(X) = \mathbb{E}[Y \mid X, E = e]$ for the conditional mean under execution on record X, and averaging over the known behavior distribution gives

$$\mathbb{E}_{E,Y|X} \Psi = \sum_e d_e \widehat{z}_e + \sum_e b_e \frac{d_e}{b_e} (z_e - \widehat{z}_e) = \sum_e d_e z_e.$$

Thus the correction is conditionally unbiased for a fixed policy comparison when each residual has conditional mean $z_e(X)$ on its recorded trace and the behavior probability is known. With record inclusion probability s_i, the sampled residual receives weight w_i / s_i, as in Eq. (9). The estimator uses the recorded behavior probabilities and full inverse-probability weights. Its variance depends on these sampling probabilities. Statistical sample size is determined by independent reference requests, with all starts, seeds, and timing repeats from one request retained in the same cluster.

For a finite weighted reference panel, write $\widehat{\Delta}_\theta^{\mathrm{model}} = \sum_i w_i \sum_e [q_\theta(e \mid X_i) - q_0(e \mid X_i)] \widehat{z}_{ie}$. If record i is included with known probability $s_i > 0$ and its action E_i is sampled with probability b_{iE_i}, the corrected gradient is

$$\nabla_\theta \widehat{\Delta}_\theta^{\mathrm{model}} + \sum_{i \in \mathcal{I}} \frac{w_i}{s_i} \frac{\nabla_\theta q_\theta(E_i \mid X_i)}{b_{iE_i}} (Y_i - \widehat{z}_{iE_i}).$$

Differentiating the estimator requires sampling support $b_{ie} > 0$ for every action with $\nabla_\theta q_\theta(e \mid X_i) \neq 0$, including parameter points where $q_\theta - q_0 = 0$. Freezing the panel, model values, and collection probabilities before feedback gives, at every prespecified parameter point, expectation $\sum_i w_i \sum_e \nabla_\theta q_\theta(e \mid X_i) z_{ie}$. This follows by the same inclusion and action expectation used above. With integrable derivatives it also differentiates the population objective. Training optimizes the collected empirical objective; independent confirmation freezes the selected configuration before its new evaluation samples.

A.3 Common-threshold relaxation and error bounds

The finite address domain $\mathcal{U}$ is the shared candidate supertree defined in the problem setup; it specifies search addresses, while each reference state realizes its own tokens and path scores. We evaluate the frozen reach and stop-action probabilities q^0, R^0 on complete draft traces generated under the current draft structure. Candidate-node score marginals and target-side contributions come from traces that cover $\mathcal{U}$, start from the same reference states, and are paired by start state. The shared prefix is checked for draft outputs, processed target probabilities, and cache semantics. A changed generation layout receives updated probability records and latency calibration for its affected prefixes.

For each action-conditioned distribution, let $Q_u \in [0, 1]$ be a candidate node's path score with mean μ_u and across-state variance v_u. For $0 < \mu_u < 1$ and $0 < v_u < \mu_u(1 - \mu_u)$, the moment-matched Beta marginal has parameters

$$c_u = \frac{\mu_u(1 - \mu_u)}{v_u} - 1, \qquad a_u = \mu_u c_u, \qquad b_u = (1 - \mu_u)c_u.$$

For zero-variance scores, the marginal is a point mass at its mean. For a score with variance $\mu_u(1-\mu_u)$, its support is zero and one, with masses $1 - \mu_u$ and μ_u. These distributions are represented by their discrete masses. At a threshold atom, use one common $\tau \in [0, 1]$ for the competing nodes and define $p_u = \Pr(Q_u > t) + \tau \Pr(Q_u = t)$. Choose it so that the expected retained count equals K. Equivalently, draw an independent Bernoulli variable with mean τ and include all threshold-tied nodes when it is one. This coupling preserves the nesting with hard top-K on every realization. Continuous marginals recover $p_u = \Pr(Q_u > t)$. Fitted grid distributions specify their finite proxy and certificate.

Statistics under ν_{jh} use $w_{jh} > 0$. An action with zero stop probability retains its structural legality constraints. Its feasibility coefficients use a prespecified supported reference distribution. Address availability enters the feasibility mask before fitting. Valid population moments lie in $0 \leq v_u \leq \mu_u(1-\mu_u)$; fitted moments and the chosen family are stored with the candidate. Across-state variance describes score variation; independent reference requests govern estimation precision.

The common threshold relaxes the per-state cardinality constraint. For fixed candidate nodes and integer budget K, let $Z_u \in [0, 1]$ be a relaxed retention weight. The online rule retains exactly K nodes in every state, whereas the relaxation requires only $\sum_u \mathbb{E}Z_u = K$. For any threshold t,

$$Q_u Z_u = tZ_u + (Q_u - t)Z_u \leq tZ_u + (Q_u - t)_+,$$

which yields

$$\mathbb{E}\sum_u Q_u Z_u \leq Kt + \sum_u \mathbb{E}(Q_u - t)_+. \tag{16}$$

With continuous marginals and $\sum_u \Pr(Q_u > t) = K$, the rule $Z_u = \mathbf{1}\{Q_u > t\}$ meets the expected budget and attains the bound. At an atom, the common tie mixture also attains it: the additional term $(Q_u - t)Z_u$ vanishes on tied nodes, while $\sum_u \mathbb{E}Z_u = K$ remains satisfied. Thus the common-threshold solution optimizes retained path-score mass under the relaxed budget; target-side contribution enters afterward through $\bar{g}_{jhu}$.

Under the action-conditioned distribution, let I_u^K indicate statewise top-K selection, $s_u = \mathbb{E}I_u^K$, p_u be its threshold-inclusion probability (with the common tie fraction at atoms), and $\bar{g}_u = \mathbb{E}g_u$. For a fixed generated draft structure,

$$\mathbb{E}[I_u^K g_u] - p_u \bar{g}_u = (s_u - p_u)\bar{g}_u + \mathrm{Cov}(I_u^K, g_u). \tag{17}$$

The first term is the hard-selection versus threshold-inclusion gap at a fixed node value; the covariance is the state-dependent contribution omitted by the product of marginals. For a frozen proposal, paired complete draft traces from the same starts estimate both terms. With normalized action-conditioned reference weights α_i, define

$$\widehat{s}_u = \sum_i \alpha_i I_{iu}^K, \qquad \widehat{g}_u = \sum_i \alpha_i g_{iu}, \qquad \widehat{c}_u = \sum_i \alpha_i (I_{iu}^K - \widehat{s}_u)(g_{iu} - \widehat{g}_u).$$

The empirical difference is exactly $\sum_i \alpha_i I_{iu}^K g_{iu} - \widehat{p}_u \widehat{g}_u = (\widehat{s}_u - \widehat{p}_u)\widehat{g}_u + \widehat{c}_u$. Here α_i is proportional to the reference weight times $q_{jh}^0(X_i)$, and $\widehat{p}_u$ comes from the fitted threshold distribution. Independent paired evaluation then measures the complete candidate effect after policy adaptation.

For fixed node values $r_u = \bar{g}_u$, the following bound quantifies the first term. Fix a generated candidate tree and use its true score marginals. Let p_u be the common-threshold inclusion probability, including the common tie mixture when required, and s_u the probability that node u is selected by statewise top-K. Let S_p contain the K largest p_u, and define $b(p) = \sum_{u \notin S_p} p_u$. Then

$$\left| \sum_u r_u s_u - \sum_u r_u p_u \right| \leq (\max_u r_u - \min_u r_u) b(p). \tag{18}$$

To see this, couple the two rules on the same score realization and use a consistent tie rule. The hard top-K set and the threshold set, with all threshold ties included or excluded together, are nested on each realization, so their indicator vectors differ in ℓ_1 norm by $|N_t - K|$, where N_t is the size of that threshold set. Let A_t count retained threshold nodes outside S_p and B_t count excluded threshold nodes inside it. Then $N_t - K = A_t - B_t$ and $\mathbb{E}A_t = \mathbb{E}B_t = b(p)$, giving $\|s - p\|_1 \leq \mathbb{E}|N_t - K| \leq 2b(p)$. Since p and s both sum to K, centering r_u at the midpoint of its range gives Eq. (18). The proof allows dependence between node scores. Equation (17) additionally accounts for selection–contribution covariance; fitted score tails and predicted latency enter the structural proxy below.

To interpret the value of adding an address, hold score marginals and node values fixed and locally relax x_u. Let $p_u(t) = 1 - F_u(t)$ and assume $D(t) = \sum_u x_u f_u(t) > 0$. Suppressing action (j, h), the proxy sensitivity is

$$\begin{aligned} \mathrm{d}\widetilde{A}_{jh} &= r_\partial \, \mathrm{d}K + \sum_u p_u(t)(\bar{g}_u - r_\partial) \, \mathrm{d}x_{ju}, \\ r_\partial &= \frac{\sum_u x_{ju} \bar{g}_u f_u(t)}{\sum_u x_{ju} f_u(t)}. \end{aligned} \tag{19}$$

The first term values an additional verification slot; the second compares a candidate with the threshold value it may displace. To derive the expression, differentiate the expected-budget equation:

$$\sum_u p_u(t) \, dx_u - D(t) \, dt = dK.$$

For $\widetilde{A} = 1 + \sum_u x_u r_u p_u(t)$,

$$d\widetilde{A} = \sum_u r_u p_u(t) \, dx_u - \sum_u x_u r_u f_u(t) \, dt.$$

Substituting the budget differential gives Eq. (19). If $K = \alpha \sum_u x_u$, the marginal value of adding node u at fixed α is $p_u(t)(r_u - r_\partial) + \alpha r_\partial$. This is a continuous local sensitivity; the binary structure is evaluated by the finite optimization.

For numerical solution, threshold tails are tabulated on a common ordered grid. At a score atom, consecutive vertices store the strict-tail vector and the tail vector including its atom; interpolation between these vertices represents the common tie fraction τ. Within a continuous interval it represents the fitted tail interpolation. Shared nonnegative SOS2 weights $\lambda_{jh\ell}$ sum to one, so every node competing under a verification-size option uses the same threshold. With $p_{jhu} = \sum_\ell \lambda_{jh\ell} p_{jhu}(t_\ell)$, binary-product constraints encode $y_{jhu} = x_{ju} p_{jhu}$. Lookup constraints map draft structure size to $K_h(n_j)$. Physical layout constraints apply to the candidate draft structure and the statewise verification trees produced by hard top-K pruning; the soft weights y_{jhu} describe expected inclusion; actual verification executes the legal hard-pruned trees. Ancestry, ranked-sibling closure, cross-decision point nesting, and legal support complete the finite domain. Zero-use actions retain execution support even though their objective weight is zero. Degenerate score masses, costs, and device features must have exact finite encodings before the solver certificate applies.

For explicit finite encoding, set $x_{ju} = \mathbf{1}\{u \in U_j\}$ for non-root addresses, and treat the root as always present. If $p(u)$ is a non-root parent and u^- the preceding ranked sibling, impose

$$x_{ju} \leq x_{j,p(u)}, \qquad x_{ju} \leq x_{j,u^-}, \qquad x_{ju} \leq x_{j+1,u}.$$

The first two constraints apply where the indicated address exists. Depth masks set unavailable addresses to zero; candidate support and hardware limits add finite constraints for every enabled verification-size option. For $0 \leq p_{jhu} \leq 1$, the exact binary-product encoding is

$$0 \leq y_{jhu} \leq x_{ju}, \qquad y_{jhu} \leq p_{jhu}, \qquad y_{jhu} \geq p_{jhu} - (1 - x_{ju}).$$

One-hot size selectors encode $n_j = \sum_u x_{ju}$ and the verification-size option lookup $K_h(n_j)$. The expected-budget equation is $\sum_u y_{jhu} = K_h(n_j)$. A draft path score multiplies conditional draft probabilities along its ancestor chain. Descendants therefore score at most their ancestors, and higher-ranked siblings score at least lower-ranked siblings. Pruning uses a deterministic tie rule consistent with both orders. This yields legal hard verification trees from feasible generated draft structures. Grid tails and latency lookup values are frozen coefficients in the finite solve.

Objective under frozen reach and stop-action probabilities. The distribution ν_{jh} is defined by the state-conditioned stop-action probability $q^0_{jh}(X)$, rather than by its marginal mass alone. Consequently, $\bar{g}_{jhu}$ and the score distribution are fitted under the histories in which the policy uses that action. Stage costs retain the state-conditioned reach probability $R^0_j(X)$. These coefficients are frozen for U2. After solving, the new shared draft structure supplies updated observations and action values for U1.

For the fixed reference rate ρ_{str}, a feasible incumbent with objective value L and a valid solver upper bound U satisfies

$$0 \le \max_{\Gamma \in \mathcal{G}_{\mathrm{exe}}} \widetilde{J}_{\rho_{\mathrm{str}}}(\Gamma; q^0) - L \le U - L.$$

This is the objective certificate for the U2 solve. The ratio certificate below evaluates the same fitted progress and complete time at the incumbent's own predicted rate.

Let $\widehat{A}(\Gamma)$ and $\widehat{B}(\Gamma)$ be proxy progress and complete time on the finite executable domain. Evaluate the solver gap at the candidate's own predicted rate $\widehat{\rho} = \widehat{A}(\widetilde{\Gamma})/\widehat{B}(\widetilde{\Gamma})$. If every feasible denominator is at least $B_{\min} > 0$ and the solver establishes

$$\sup_{\Gamma \in \mathcal{G}_{\mathrm{exe}}} [\widehat{A}(\Gamma) - \widehat{\rho}\widehat{B}(\Gamma)] \le \delta, \tag{20}$$

then $0 \le \widehat{\eta}^*_{\mathcal{G}_{\mathrm{exe}}} - \widehat{\rho} \le \delta / B_{\min}$. For every feasible Γ, the bound implies

$$\frac{\widehat{A}(\Gamma)}{\widehat{B}(\Gamma)} - \widehat{\rho} = \frac{\widehat{A}(\Gamma) - \widehat{\rho}\widehat{B}(\Gamma)}{\widehat{B}(\Gamma)} \le \frac{\delta}{B_{\min}}.$$

Taking the maximum proves the upper gap bound; feasibility of $\widetilde{\Gamma}$ proves its nonnegative lower bound. This certificate concerns the finite surrogate under frozen reach and stop-action probabilities and fitted statistics; independent same-population execution establishes whether the adapted configuration improves measured throughput.

A.4 Independent acceptance and multi-round stopping

For acceptance, freeze the adapted candidate c and baseline b. Evaluate both from the same saved cycle-start states on independent reference requests. If request i has starts s, let ω_{is} be their prescribed within-request weights and set

$$U_{i\xi} = \sum_s \omega_{is} G_{is\xi}, \qquad V_{i\xi} = \sum_s \omega_{is} T_{is\xi}, \qquad \xi \in \{b, c\}.$$

Request-level averaging then gives $\eta_\mu(\xi) = \mathbb{E}U_{i\xi}/\mathbb{E}V_{i\xi}$. The same weights define position selection, policy learning, structural statistics, and acceptance. Equal within-request weights give request-balanced starts; unnormalized totals give start-count-weighted starts. Complete cycle time includes every operation required by the evaluated configuration.

Use independent pilot clusters to obtain a confidence interval $[r - e_\rho, r + e_\rho]$ for the baseline rate. On paired confirmation clusters, set $d_i = V_{ic} - V_{ib}$ and $Z_i = (U_{ic} - U_{ib}) - r d_i - s V_{ic}$ for a prespecified minimum gain $s \ge 0$. Let r_Z, r_d be simultaneous confidence radii for their means. Accept when

$$\overline{Z} - r_Z - e_\rho(|\overline{d}| + r_d) > 0. \tag{21}$$

On the joint coverage event, this implies $\eta_\mu(c) > \eta_\mu(b) + s$. To verify the rule, let $\rho_b = \eta_\mu(b)$. The pilot event gives $|\rho_b - r| \le e_\rho$, and the paired event gives $\mathbb{E}Z \ge \overline{Z} - r_Z$ and $|\mathbb{E}d| \le |\overline{d}| + r_d$. Therefore

$$\mathbb{E}(U_c - U_b) - \rho_b \mathbb{E}(V_c - V_b) - s\mathbb{E}V_c \ge \overline{Z} - r_Z - e_\rho(|\overline{d}| + r_d) > 0.$$

Since $\mathbb{E}U_b = \rho_b \mathbb{E}V_b$ and $\mathbb{E}V_c > 0$, division by $\mathbb{E}V_c$ proves the claimed throughput improvement. The acceptance statement requires simultaneous coverage of the pilot rate interval and the confirmation means. Reference requests form independent clusters; all cycle starts and repeated measurements

from one request remain in the same cluster. Candidates and the reference rate are frozen before confirmation. Valid cluster-mean intervals supply r_Z, r_d, and error allocations across candidates, looks, and rounds sum to the prescribed total error probability. On this joint event, the algebra above proves the rate improvement.

Error budgets account for the number of candidates, repeated looks, and update rounds. A candidate satisfying the rule becomes the next baseline. All other comparisons retain the current baseline; additional paired samples can refine the confidence intervals.

Multi-round stopping has three outcomes. For a prespecified finite neighborhood $\mathcal{N}(b)$ of the current baseline b, the certificate requires simultaneous fixed-μ rate intervals satisfying

$$\max_{c \in \mathcal{N}(b)} U_\eta(c) - L_\eta(b) \le \epsilon_{\text{stop}}, \tag{22}$$

where $U_\eta(c)$ and $L_\eta(b)$ are upper and lower bounds on candidate and baseline cycle throughput. The certificate uses valid simultaneous bounds for every member of the prespecified neighborhood.

Operational stability is a separate stopping diagnostic. Its window freezes μ, the position choice, proposal recipe, weights, and thresholds. At each round, compare every executed proposal with the pre-update baseline, including proposals later rejected. For shared draft structures $\Gamma = (U_1, \dots, U_J)$, define the weighted edit distance by

$$D_\Gamma(\Gamma_c, \Gamma_b) = \sum_{u \in \mathcal{U}} \kappa_u \sum_{j=1}^{J} \left| \mathbf{1}\{u \in U_j^c\} - \mathbf{1}\{u \in U_j^b\} \right|,$$

where $\kappa_u > 0$ is a fixed, recorded edit weight on each variable address u (the always-present root is excluded). Because an address is counted at every decision point where its membership differs, D_Γ measures cumulative draft structure edits. To define D_π, fix a joint sampling rule for candidate and baseline draft histories that preserves each executor's marginal law from the same μ start. Use the union of their labeled terminal actions, extending disabled terminal actions by zero. Their conditional terminal-action distributions have total variation $\frac{1}{2}\sum_e |q_c(e \mid X_c) - q_b(e \mid X_b)|$, and D_π is its expectation under this recorded coupling. The reach-weighted cost change is $D_R = \mathbb{E}_\mu \sum_j \bar{c}_j |R_{c,j} - R_{b,j}|$, where $\bar{c}_j$ is the reference mean cost of stage j.

Use simultaneous upper confidence bounds U_π and U_R, together with the rate-interval bound $U_{\text{gain}} = U_\eta(c) - L_\eta(b)$ on each proposal's cycle-throughput improvement. Each proposal in each of K_{stab} consecutive rounds must satisfy $D_\Gamma \le \epsilon_\Gamma$, $U_\pi \le \epsilon_\pi$, $U_R \le \epsilon_R$, and $U_{\text{gain}} \le \epsilon_{\text{gain}}$. A stability round contains at least one proposal generated by the fixed recipe; a repeated configuration records zero change after that recipe has been executed. Unresolved intervals reset the window and may trigger more fixed-candidate samples. Reaching a round or data limit is a budget exit. Operational stability records whether draft structure edits, policy shifts, reach-weighted costs, and bounded cycle-throughput gains remain within prespecified tolerances over consecutive rounds; the finite-neighborhood test separately bounds the evaluated alternatives relative to the baseline.

Full-request evaluation follows the subsequent states induced by each deployed configuration and measures committed tokens per wall-clock time. The fixed-μ comparison above evaluates the conditional updates on their shared reference population.

B Policy Observations and Conditional Updates

B.1 Observations and legal action probabilities

At decision point j, the observation O_j contains the confirmed prefix, candidates and draft scores generated so far, context length, remaining output allowance, and decoding settings. It is measurable with respect to the information $\mathcal{F}_j$ in Equation (4). Later candidates and target verification outcomes enter the offline action labels.

Let $\mathcal{H}_j(X)$ be the available verification-size options on trace X. The policy gives a continuation probability $a_j(O_j)$ and a conditional distribution $\beta_{jh}(O_j)$ over $h \in \mathcal{H}_j(X)$. Unavailable actions have probability zero, and available stop probabilities sum to one. The terminal decision point has $a_J = 0$. Equation (8) then gives

$$\sum_h q_{jh}(X) = R_j(X) - R_{j+1}(X), \qquad \sum_{j,h} q_{jh}(X) = 1.$$

Thus every trace terminates in one legal action. The root-only action handles immediate EOS or a single remaining output token. Feasibility masks provide a legal fallback for every enabled decision point and reference trace.

The policy parameterization maps O_j to these probabilities. A masked softmax over the available actions is one finite differentiable realization. The theory uses the induced terminal-action distribution q_θ, allowing the observation representation and maximum executable depth to follow the chosen draft structure.

B.2 Policy gradients and adaptation

For fixed records, action net-gain estimates, and reference rate, the full-action model term has gradient

$$\nabla_\theta \sum_i w_i \sum_e \big[q_{\theta,e}(X_i) - q_{\theta_0,e}(X_i)\big]\widehat{z}_{i,e} = \sum_i w_i \sum_e \nabla_\theta q_{\theta,e}(X_i)\widehat{z}_{i,e}.$$

Appendix A.2 derives the same-trace residual correction and its conditional expectation. Optimization combines this objective with the average terminal-action KL constraint in Section 4.3. A finite KL radius keeps the updated distribution on the baseline's terminal-action support; collection probabilities cover every action with a nonzero policy derivative.

After a U2 update, records and action net-gain estimates are evaluated on the new shared draft structure. U1 adapts the policy to its available observations and actions. Independent paired evaluation then compares the adapted structure–policy pair with the current baseline. Appendix A.4 gives the gain criterion and the three termination events: a neighborhood gain bound, operational stabilization, or budget exhaustion.

C Verification and Expected Progress

C.1 Verification assumptions

The progress formula conditions on a cycle state S and a realized draft history ω. Thus each rank address denotes a particular token and token prefix. The following assumption specifies when probabilities recorded on a coverage tree can be reused for its verification subtrees.

Assumption 1 (Preserved prefixes and ordered residual verification)**.** *Every admissible subtree is ancestor closed and contains distinct sibling tokens. Its verifier tests siblings in the prescribed order. At a retained token prefix, pruning other branches preserves the processed target distribution p_κ. After rejecting a candidate, the verifier removes that token's probability mass and renormalizes the residual distribution. Conditional on the full draft record and preceding verification history, each test uses a fresh uniform variable and compares it with the corresponding normalized residual target mass. The controller selects the verification tree before these variables are sampled. The root and termination conventions are the same for all compared trees.*

Under Assumption 1, each retained prefix has the same processed target probability in the coverage tree and its legal verification subtrees. These probabilities give the additive progress coefficients below. Execution-specific probabilities can also be recorded directly for a calibrated layout.

C.2 Residual acceptance and termination

Fix a parent prefix and let $x_1, \dots, x_H$ be its ordered, distinct candidates. Write $p_h = p_\kappa(x_h \mid \text{prefix})$ and $s_h = \sum_{g<h} p_g$. Let F_h be the event that the first $h-1$ candidates are rejected. We have $\Pr(F_1) = 1$. If $s_h < 1$, residual verification gives

$$\Pr(x_h \text{ accepted} \mid F_h) = \frac{p_h}{1 - s_h}, \qquad \Pr(F_{h+1}) = (1 - s_h)\left(1 - \frac{p_h}{1 - s_h}\right) = 1 - s_{h+1}. \quad (23)$$

Induction therefore yields $\Pr(F_h) = 1 - s_h$ and $\Pr(x_h \text{ accepted}) = (1 - s_h)p_h/(1 - s_h) = p_h$. If $s_h = 1$, the event F_h has probability zero. All remaining distinct candidates have zero mass, and their branches are unreachable; the verifier handles the branch as unreachable.

If all H candidates are rejected and $\sum_h p_h < 1$, the remaining token is sampled from the target distribution restricted to their complement. Its unconditional mass at any untested token x is

$$\left(1 - \sum_h p_h\right) \frac{p_\kappa(x \mid \text{prefix})}{1 - \sum_h p_h} = p_\kappa(x \mid \text{prefix}).$$

Together with the candidate masses above, this recovers the processed target distribution at each visited prefix. When the initial root follows that same target rule, induction over committed prefixes establishes distribution preservation for any legal tree and any controller choosing its tree from pre-verification observations. Generation stops according to the common output allowance and EOS convention.

For a non-root node u, multiply these conditional probabilities along its retained parent chain. With $r_o = 1$, its acceptance probability is

$$r_u = \prod_{v \in \text{path}(u) \setminus \{o\}} p_\kappa(x_v \mid \text{prefix}(v)). \quad (24)$$

The product follows from the chain rule. Acceptance events for siblings are mutually exclusive.

The root o is the cycle's first token, selected before drafting and committed in the current cycle. Let its depth be zero and let $n_{\text{rem}} \geq 1$ be the remaining output allowance. We count an end-of-sequence token itself and exclude its descendants. Define

$$m_u = \mathbf{1}\{d(u) + 1 \leq n_{\text{rem}}\}\mathbf{1}\{\text{every strict ancestor of } u \text{ is not EOS}\}. \quad (25)$$

In the main text, $C^\circ = \{u \in C \setminus \{o\} : m_u = 1\}$. Thus summing over C° is identical to using $m_u r_u$ over all non-root nodes. The committed count is one for the root plus the indicators of accepted, eligible non-root nodes. Linearity of expectation proves (1). If the root is EOS, all descendant masks vanish and the count is one. If $n_{\text{rem}} = 0$, no cycle starts and the count is zero. An immediately terminating root is handled by its own terminal action rather than by the ordinary multi-action draft process.

For $C_1 \subseteq C_2$ under the same probability field and termination convention,

$$\bar{G}(C_2) - \bar{G}(C_1) = \sum_{u \in C_2 \setminus C_1} m_u r_u \geq 0. \quad (26)$$

The corresponding throughput comparison divides this increased progress by the complete time of each tree. If I_u records whether a coverage-tree verification accepted u, then $1 + \sum_{u \in C \setminus \{o\}} m_u I_u$ is a conditionally unbiased sample of $\bar{G}(C)$. Separate subtree executions sample from their own conditional verification histories; probability labels give the shared conditional expectation. The selected subtree is fixed conditional on the draft record before verification randomness is sampled. For $Y = G - \rho T$, its conditional mean is $\bar{G} - \rho \bar{T}$, and the law of total variance gives

$$\text{Var}(Y) = \text{Var}(\mathbb{E}[Y \mid X]) + \mathbb{E}\,\text{Var}(Y \mid X).$$

This establishes the variance reduction from exact conditional averaging of complete net gain. Fitted values enter the residual correction in Appendix A.2.

C.3 Rank support across states

A shared tree selects rank addresses, whose token realizations differ across states. Let $\mathcal{U}_{i,j}$ be the addresses actually generated by decision point j in history i, and let $\mathcal{I}_e$ contain the histories on which

terminal action $e = (j, h)$ is enabled. The shared cumulative draft structure at decision point j uses a shared set U_j satisfying

$$U_j \subseteq \bigcap_{i \in \mathcal{I}_e} \mathcal{U}_{i,j}, \qquad C_e(X_i) \setminus \{o\} \subseteq U_j \cap \mathcal{U}_{i,j}. \tag{27}$$

The generated addresses are shared across reference histories; the hard-pruned verification tree $C_e(X_i)$ varies with the scores recorded in each history. Every enabled terminal action has legal support, including terminal actions assigned zero probability by the current policy. Availability masks describe generated candidates, while termination masks describe committed tokens. A fixed-shape generator pays for generated candidates even when EOS or the output allowance excludes their descendants from commitment. Early generation termination instead enters the recorded support and action masks.

The posterior portfolio in Section 3 uses the state-specific domain $\mathcal{D}_i$ and requires a supported member on every positively weighted state. A common executable fallback can ensure this condition, provided it is allowed by the verifier. In the structural update, masks and terminal actions are fixed before solving, and (27) is imposed on every retained terminal action. A different source draft structure changes token realizations and requires new histories and coefficients.

C.4 Finite-domain shared trees and portfolios

For a recorded history i, let $g_{iu} = m_{iu} P_{iu}$ be the target-side contribution of address u. For a legal shared tree C, Equation (1) gives $\bar{G}_i(C) = 1 + \sum_{u \in C \setminus \{o\}} g_{iu}$. A latency signature specifies the finite features used by the prediction model. For a fixed signature, its denominator $\widehat{T}_i(C) = t_i > 0$ is fixed on each history. The shared-tree objective is therefore

$$\sum_i w_i \widehat{\eta}_i(C) = \sum_i \frac{w_i}{t_i} + \sum_{u \in C \setminus \{o\}} \sum_i \frac{w_i g_{iu}}{t_i}.$$

Binary address variables, ancestry and ranked-sibling constraints, support masks, and the prescribed signature form a finite integer program. Solving every feasible signature and comparing the objective values gives the optimum of this weighted historywise-ratio objective in the finite supported domain.

For a portfolio $\mathcal{P} = \{C_1, \ldots, C_K\}$, assign each history to a supported member maximizing its posterior surrogate rate. With these assignments fixed, member C_k solves the shared-tree problem with weights $w_i \mathbf{1}\{i \text{ is assigned to } k\}$. Reassignment and exact member updates each preserve or increase Equation (3). A deterministic tie rule and retention of an incumbent at equal objective value give termination when the finite assignment and member choices stabilize. The portfolio objective averages historywise surrogate ratios; Equation (4) evaluates the joint ratio of expected progress to expected complete time.

D Pre-Drafting Routing and Drafting-Time Information

D.1 Posterior portfolios and pre-drafting routing

The routing study compares a learned pre-drafting selector with a fixed template and posterior selection from the same portfolio. It covers nine families and 71 complete router configurations, with 194 configuration–split cells: 71 training, 71 selection-validation, 44 test/development-test, and eight external results. A further 16 configurations control switching alone. Modifications change the decision model, utility objective, switch gate, or training data. Model and threshold selection use the corresponding validation split.

For the posterior portfolio illustration in Section 3.2, the four-tree and fixed-tree surrogate rates are 117.560 and 107.569 tokens/s on 50 ShareGPT test requests, and 152.730 and 143.382 tokens/s on ten HumanEval test requests. These pairs give the reported gains of about 9.3% and 6.5%. The portfolio optimizes the weighted historywise surrogate-rate objective in Equation (3).

A separate four-tree routing test contains 2,071 cycles from 50 ShareGPT requests. Posterior selection, template 0, and the learned selector attain 125.369, 106.033, and 102.178 local proxy tokens/s. This diagnostic uses candidate draft-and-verify time. The routed member differs from the posterior-best member in 1,274 cycles. Classifying those disagreements by the corresponding template pair's mean local time gives 583 slower, 402 faster, and 289 unresolved cases, as shown in Figure 1(b,c).

Six early four-way classifiers were evaluated by accuracy. Six later families contain 35 distinct family–configuration keys and 51 nontraining accuracy cells, whose maximum is 49.58%. In the 2,071-cycle test, accuracy is 38.48%. On the 1,727 cycles where the best local rate exceeds the second best by at least 5%, accuracy is 38.10%; always choosing the largest class in this subset gives 35.03%. The margin-filtered comparison therefore retains the selection difficulty seen on the full test.

Recovery of posterior gain. For each protocol, define recovery as

$$100 \times \frac{\widehat{\eta}_{\text{route}} - \widehat{\eta}_{\text{fixed}}}{\widehat{\eta}_{\text{post}} - \widehat{\eta}_{\text{fixed}}}. \tag{28}$$

Zero equals the fixed-template rate and 100 equals the posterior portfolio rate. Table 2 gives the selected configuration in each family. The gate-then-select, abstention, and hard-negative families contain the 36 paired internal-test configurations in Figure 1(a). Each has positive training recovery and negative paired test recovery. Expanded training and utility/uncertainty use their own development-test protocols and provide the positive cases in the table.

Table 2: Posterior portfolio gain recovered (%) by the selected configuration under each family's protocol. Columns distinguish selection validation, test/development test, and external evaluation.

Family	Train	Selection val.	Test/dev-test	External
Gate then select	96.23	-1.83	-34.48	—
Abstention	6.32	6.59	-9.80	—
Hard negatives	94.43	9.27	-12.29	—
Expanded K4 training	14.44	14.64	0.26	7.49
Utility/uncertainty	18.17	17.97	2.00	17.64
Residual utility	77.23	6.87	—	—
More training requests	13.07	7.41	—	—
On-policy refresh	10.30	10.99	—	—
Corrected draft state	65.25	10.28	—	—

The two external cases recover 7.49% and 17.64% of posterior gain, equivalent to local proxy-rate increases of 1.36% and 3.20% over their fixed templates. Their values complement the 2.00% development-test recovery in the utility/uncertainty family.

Switching opportunity and template identification. Holding three learned switch gates fixed, we replace each downstream selector with one using posterior verification outcomes. The resulting test recoveries are 50.28%, 75.48%, and 75.48%, versus -34.48%, -9.80%, and -12.29% for their learned selectors. Both selectors use the same gate and available templates. The difference isolates template identification among the triggered switches: posterior evidence identifies valuable choices that the pre-drafting selector misses.

D.2 Information acquired during drafting

The prediction study uses 20 historical development requests each from ShareGPT and HumanEval, with 1,451 cycle states. It predicts accepted tokens at depth 12 from the information available at earlier depths. Within each dataset, sorted request identifiers are assigned round-robin to five folds. All cycles from a request stay in the same fold. Training gives each request equal total weight; evaluation averages absolute errors within requests and then across requests.

All predictors use the same random-tree ensemble configuration. The context-only predictor receives context length. The depth-3, depth-6, and depth-9 predictors additionally receive the first 100 cumulative draft-score features available at that depth. Table 3 gives the resulting errors.

Table 3: Mean absolute error (tokens) for predicting depth-12 accepted tokens under five request-isolated folds.

Input	All requests	ShareGPT	HumanEval
Constant predictor	3.360	3.113	3.606
Context length only	3.392	3.190	3.594
Depth-3 observations	2.536	2.285	2.786
Depth-6 observations	2.188	1.946	2.429
Depth-9 observations	2.185	1.981	2.390

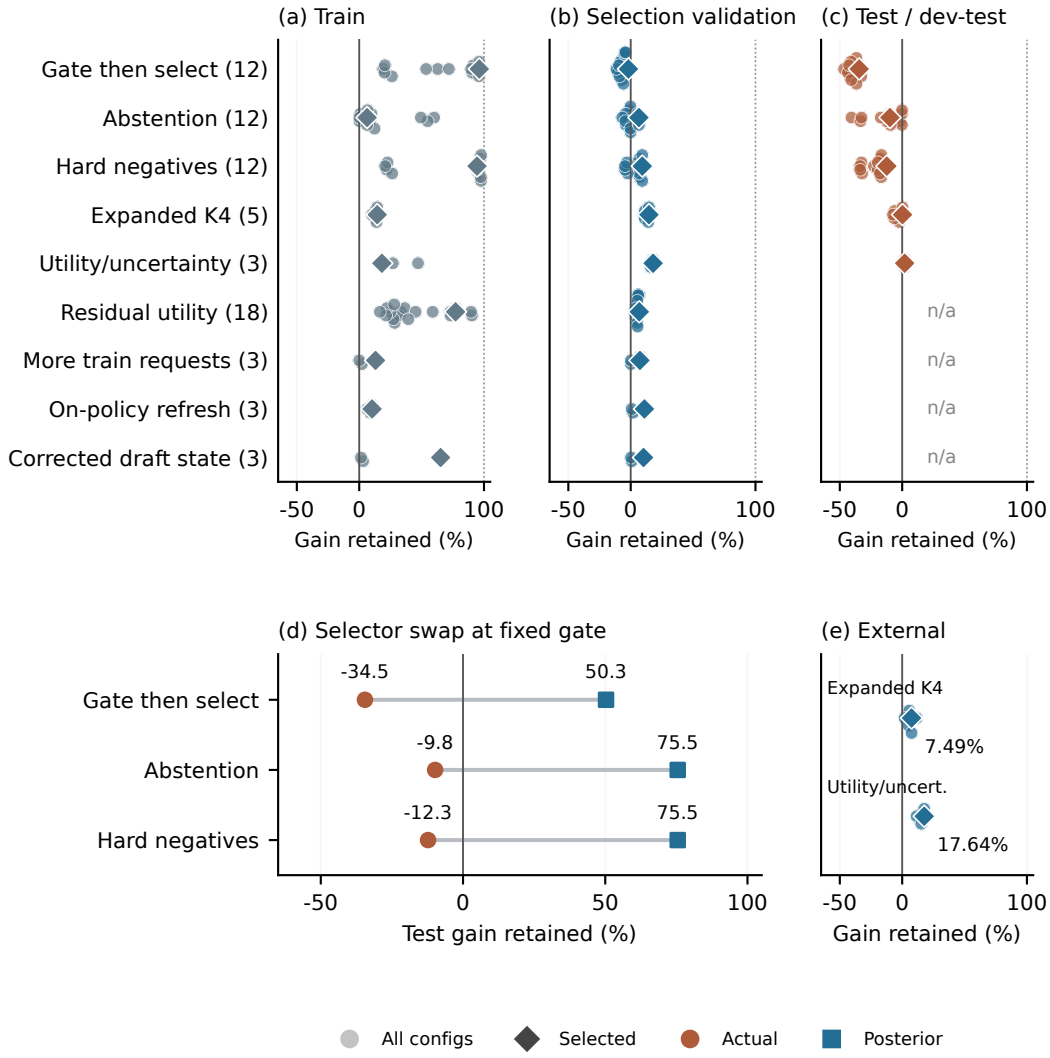


Figure 6: Pre-drafting routing across nine families. The upper panels show training, selection-validation, and test/development-test recovery for individual configurations and selected configurations. The lower panels compare learned and posterior selectors under fixed switch gates and show the two positive external cases. Recovery is defined in Equation (28); each family follows its stated protocol.

Intermediate observations lower the pooled error from 3.392 to 2.185 tokens. Most of the reduction occurs by depth 6. The successive candidate pools annotated in Figure 1(d) contain 66, 169, 264, and 356 candidates; the increments are 103, 95, and 92. These annotations describe the candidate accumulation in this prediction collection.

Local action selection. A separate probe on the same 40 requests and 1,451 common-start histories compares selectors using context length with selectors additionally using the first 100 depth-3 scores. Both choose among measured depth-3, depth-6, and depth-9 actions. Five request-isolated folds, request-equal weights, and identical regressors are fixed for this comparison. Their mean local proxy rates are 149.566 and 151.811 tokens/s; the depth-3 observations improve the rate on 34 of 40 requests. This local rate divides accepted tokens by draft, verification, and fixed-gate time. Complete-cycle time in Equation (2) additionally charges the controller, commitment, and cache operations. The local comparison supplies a decision-value diagnostic for intermediate observations.

D.3 Shared computation and marginal progress

On the historical staged draft structure, 11,698 adjacent-stage transitions retain generated candidates and append a suffix. The cumulative counts at depths 4, 7, 10, and 12 are 154, 260, 333, and 405;

the remaining suffixes after the first three depths contain 251, 145, and 72 candidates. These counts describe reuse of the generated prefix across later actions.

In 141 same-state full-tree comparisons, the mean additional accepted progress from depths 4 to 7, 7 to 10, and 10 to 12 is 0.9645, 0.2057, and 0.0355 tokens per cycle. The corresponding 95% request-cluster bootstrap intervals are $[0.6522, 1.2986]$, $[0.1042, 0.3191]$, and $[0.0071, 0.0709]$. Later intervals offer less additional progress on average. The stopping comparison weighs this progress against the complete incremental action cost, as in Appendix A.1.

Historical online comparison. The historical policy can stop at depths 3, 6, and 9 and otherwise terminates at depth 12. On 20 ShareGPT and 20 HumanEval requests, with at most 1,024 new tokens and three timing repeats, its batch throughput is 133.756 tokens/s, versus 133.681 for fixed depth 9. The corresponding request-mean rates are 140.267 and 141.117 tokens/s. Batch throughput divides total output tokens by total wall-clock time within a run and then averages run rates; request-mean throughput averages individual request rates. These closely matched rates motivate the joint optimization of available verification choices and the policy that selects them.

E Additional Experimental Results

The complete four-workload throughput matrix appears below. The final column is the unweighted mean of QA, GSM8K, Alpaca, and MT-Bench.

Table 4: Complete four-workload throughput matrix by model, device, temperature, and method. Each number is the mean token throughput for the named task; the final column averages the four task values.

Model	Device/T	Method	QA	GSM8K	Alpaca	MT-Bench	Mean
Llama-8B	910B/$T=0$	EAGLE-3	77.0	101.4	97.3	97.4	93.3
Llama-8B	910B/$T=0$	LTD	80.0	103.6	102.5	101.2	96.8
Llama-8B	910B/$T=0$	**OGSPO**	83.7	112.0	108.7	105.9	102.6
Llama-8B	910B/$T=1$	EAGLE-3	57.4	86.3	76.4	69.3	72.4
Llama-8B	910B/$T=1$	LTD	57.7	84.9	78.2	68.3	72.2
Llama-8B	910B/$T=1$	**OGSPO**	62.5	93.4	82.5	74.2	78.1
Llama-8B	H800/$T=0$	EAGLE-3	110.1	137.0	143.1	135.3	131.4
Llama-8B	H800/$T=0$	GS	116.7	148.4	153.6	145.6	141.1
Llama-8B	H800/$T=0$	LTD	115.6	142.3	152.0	143.4	138.3
Llama-8B	H800/$T=0$	**OGSPO**	119.8	153.3	159.1	149.1	145.4
Llama-8B	H800/$T=1$	EAGLE-3	80.9	117.6	114.4	95.2	102.0
Llama-8B	H800/$T=1$	GS	83.1	112.8	118.9	92.9	101.9
Llama-8B	H800/$T=1$	LTD	81.0	115.8	116.8	93.6	101.8
Llama-8B	H800/$T=1$	**OGSPO**	89.4	127.7	125.0	101.3	110.8
Llama-8B	RTX 5000/$T=0$	EAGLE-3	70.4	94.5	93.0	90.7	87.2
Llama-8B	RTX 5000/$T=0$	LTD	73.2	96.6	97.9	94.2	90.5
Llama-8B	RTX 5000/$T=0$	**OGSPO**	75.3	104.2	100.9	100.1	95.1
Llama-8B	RTX 5000/$T=1$	EAGLE-3	52.2	79.8	75.2	62.9	67.5
Llama-8B	RTX 5000/$T=1$	LTD	52.4	78.5	76.9	61.9	67.4
Llama-8B	RTX 5000/$T=1$	**OGSPO**	56.2	83.9	79.4	66.3	71.4
Llama-8B	Spark/$T=0$	EAGLE-3	38.8	48.5	50.9	47.2	46.3
Llama-8B	Spark/$T=0$	LTD	40.3	49.6	53.6	49.0	48.1
Llama-8B	Spark/$T=0$	**OGSPO**	41.0	52.7	54.8	50.5	49.8
Llama-8B	Spark/$T=1$	EAGLE-3	28.3	42.1	40.1	33.3	36.0
Llama-8B	Spark/$T=1$	LTD	28.4	41.4	41.0	32.8	35.9
Llama-8B	Spark/$T=1$	**OGSPO**	30.8	43.4	41.7	34.8	37.7
Qwen-32B	H800/$T=0$	EAGLE-3	28.2	40.9	27.9	35.5	33.1
Qwen-32B	H800/$T=0$	GS	30.2	47.0	33.5	37.9	37.2
Qwen-32B	H800/$T=0$	**OGSPO**	32.4	50.1	35.2	40.3	39.5

Model	Device/T	Method	QA	GSM8K	Alpaca	MT-Bench	Mean
Qwen-32B	H800/$T=1$	EAGLE-3	28.7	43.9	31.1	34.2	34.5
Qwen-32B	H800/$T=1$	GS	28.5	43.9	31.0	34.0	34.4
Qwen-32B	H800/$T=1$	**OGSPO**	32.6	47.9	35.9	37.5	38.5
Vicuna-13B	H800/$T=0$	EAGLE-3	105.5	133.2	130.1	136.5	126.3
Vicuna-13B	H800/$T=0$	GS	105.0	137.6	134.4	142.0	129.7
Vicuna-13B	H800/$T=0$	LTD	107.6	139.3	137.2	145.0	132.2
Vicuna-13B	H800/$T=0$	**OGSPO**	110.4	144.2	139.7	145.4	134.9
Vicuna-13B	H800/$T=1$	EAGLE-3	91.7	104.8	99.6	109.4	101.4
Vicuna-13B	H800/$T=1$	GS	93.1	108.2	102.4	110.2	103.5
Vicuna-13B	H800/$T=1$	LTD	94.4	109.2	107.2	118.2	107.3
Vicuna-13B	H800/$T=1$	**OGSPO**	95.8	118.4	107.5	117.6	109.8

Table 5: Fifth-workload extensions: CNN/DailyMail on H800 and summarization on Spark GB10. These device-specific results are separate from the four shared workloads.

Setting	Workload	EAGLE-3	LTD	**OGSPO**
Llama-8B / H800 / $T=0$	CNN/DM	116.8	121.1	125.2
Llama-8B / H800 / $T=1$	CNN/DM	88.6	93.0	103.3
Llama-8B / Spark / $T=0$	SUM	38.5	–	41.1
Qwen-32B / H800 / $T=0$	CNN/DM	26.5	–	30.0
Qwen-32B / H800 / $T=1$	CNN/DM	26.5	–	29.3
Vicuna-13B / H800 / $T=0$	CNN/DM	127.3	135.0	136.9
Vicuna-13B / H800 / $T=1$	CNN/DM	103.0	110.0	108.9

E.1 Definitions for the controlled analyses

Structure and policy adaptation. Figure 3 compares four pairs within each group: original structure/original policy, new structure/frozen policy, new structure/adapted policy, and original structure/extra-trained policy. Each throughput ratio uses the original pair as its denominator. Cross-evaluation uses the same candidate trees and policy snapshots to identify best responses on each axis. For a measured rate $\eta(\Gamma, \pi)$, the sum of unilateral gaps is

$$\big[\max_{\Gamma'} \eta(\Gamma', \pi) - \eta(\Gamma, \pi)\big] + \big[\max_{\pi'} \eta(\Gamma, \pi') - \eta(\Gamma, \pi)\big],$$

where both maxima use the displayed finite candidate sets. A mutual best response has zero gap. The search panel compares a feasible fitted objective with its finite-domain optimum and valid upper bound. The proxy panel compares predicted and hard-evaluated gains on the same candidate set.

Feedback and offline learning cost. In Figure 4, relative efficiency is throughput divided by the shared reference throughput. For target r and budget b, the attainment fraction is the number of the 16 matched runs reaching r by b, divided by 16. The plotted frontier marks fraction $1/2$, and its color compares the residual-corrected method with the indicated reference at the same target and budget. Gradient agreement is the cosine between estimates from two half-batches. Signal-to-noise ratio compares gradient magnitude with variation across gradient estimates under the plotted protocol. Marginal return is the change in relative efficiency divided by the additional normalized offline budget.

Request time and conditional stopping. For each matched request in Figure 5, the time ratio is $T_{\mathrm{LTD}}/T_{\mathrm{OGSPO}}$. An occupied length bin averages these request ratios; QA and HumanEval share a color scale. At drafting step j, conditional exit probability is

$$\frac{\text{number of episodes terminating at step } j}{\text{number of episodes reaching step } j}.$$

The pooled curve aggregates the numerator and denominator across task runs. It therefore measures the decision among episodes still drafting, while q_{jh} in Equation (8) is the joint probability of reaching a decision point and stopping with option h.