HARP: HORIZON-AWARE REPRESENTATION PREFERENCE FOR PARALLEL SPECULATIVE DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Speculative decoding (SD) accelerates large language model (LLM) inference by letting a lightweight drafter propose tokens that the target model verifies in parallel. Parallel drafters push this paradigm further by predicting an entire block of future tokens from a single target computation at the verified prefix, yet they treat all positions in the block uniformly, fusing target hidden states into one shared representation and ranking candidates by token confidence. We identify prediction horizon, the distance of a position from the verified prefix, as a key factor in both drafting and verification. On the drafting side, the target layer with the strongest predictive signal shifts systematically from upper layers toward intermediate layers as the horizon grows, so different future positions favor different target representations. On the verification side, we prove that for repair trees preserving the main draft chain, the additional accepted length decomposes exactly at the first mismatch into prefix survival, alternative-token probability, and recoverable continuation, showing that the value of a repair branch depends on more than per-token confidence. Based on these results, we propose HARP (Horizon-Aware Representation Preference), a training-free inference framework that assigns each horizon its preferred combination of already-computed target layers and allocates repair branches by this reachable value. On Qwen3-4B with a frozen DFlash drafter, HARP improves end-to-end throughput by 19.7% over chain verification, compared with 9.7% for confidence-based allocation, and achieves higher throughput than recent draft-tree methods while verifying fewer nodes. A factorial study shows that the two components provide complementary gains, and HARP improves throughput by 10.8–14.3% across six target–drafter pairs. Prediction horizon thus provides a principled axis for exploiting target representations and verification structure in training-free SD acceleration.

1 INTRODUCTION

Large language models (LLMs) generate one token per forward pass, so their decoding latency is dominated by sequential, memory-bound calls to the full model. Speculative decoding (SD) relieves this bottleneck by letting a cheap drafter propose future tokens that the target LLM verifies in a single forward pass while preserving its output distribution (Leviathan et al., 2023; Chen et al., 2023). Parallel drafters such as DFlash and DFlare push this idea furthest: from target hidden states at the verified prefix they emit an entire block of H future tokens in one call (Chen et al., 2026; Zhang et al., 2026a). Because the target states of the future positions are computed only after verification, every position of the block is predicted from the *same* present computation. Current drafters fuse several target layers into one representation and share it across the block, implicitly assuming that a token one step ahead and a token fifteen steps ahead are best served by the same view of the present.

We find that different positions of the block prefer different target layers, and that exploiting this preference yields system-level gains. When we fit identical probes from every target layer to every future horizon, the lowest-loss layer drifts from the top of the network at horizon 1 toward intermediate layers at far horizons, as shown in Figure 1, and this pattern holds across tasks and target models. When we remove a single intermediate source layer from the DFlash drafter, its accuracy drops by 17.1 points of near-horizon but 28.8 points of far-horizon accuracy. A speculative block therefore has

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.

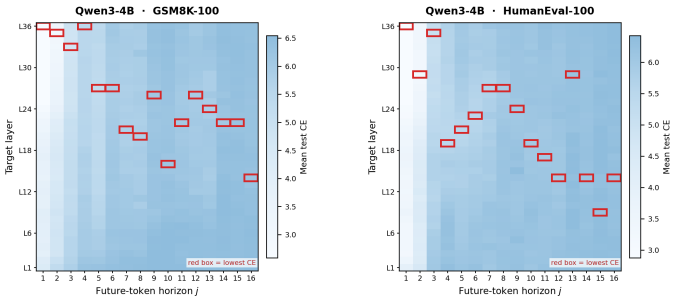

Figure 1: **The best target layer depends on the prediction horizon.** Held-out cross-entropy of identical probes that read target layer ℓ of Qwen3-4B at the verified prefix and predict the token j steps ahead, on the GSM8K-100 probe set on the left and the HumanEval-100 probe set on the right. Red boxes mark the lowest-loss layer per horizon; horizon j counts from the root token, so $j = k + 1$. Near horizons prefer the top layers, while far horizons prefer intermediate layers. Maps for Llama-3.1-8B and Qwen3-8B are in Appendix D.

a temporal structure in which positions prefer different representations, and shared fusion erases this structure before the drafter emits a token.

Faster decoding, however, requires more than better token prediction. The verifier commits only a prefix, so a correct token contributes only if every earlier token is also correct, and Proposition 1 shows that cross-entropy (CE) and accepted length can rank two predictors in opposite orders. We therefore analyse what a candidate is worth to the verifier. For the family of tree repairs that keep the main chain and attach single-alternative branches, the extra accepted length has an exact decomposition at the *first mismatch*: a branch increases accepted length only if the chain survives up to it, its alternative is the target token, and the following draft tokens continue to match, as stated in Theorem 2. We further prove in Proposition 3 that this value depends on the joint law of the continuation beyond per-token marginals, which explains why reachability-aware allocation outperforms confidence-based tree construction.

We propose HARP (**H**orizon-**A**ware **R**epresentation **P**reference), which combines the two findings into a training-free, inference-time policy for parallel drafters. A *representation policy* gives each horizon group its preferred composition of already-computed target layers, changing which candidates the frozen drafter proposes. A *verification policy* spends a fixed branch budget on the horizons with the largest reachable value. Both policies keep all model weights frozen and reuse the target computation that is already available. Figure 2 gives an overview.

Contributions. (i) We identify a horizon-dependent representation preference in parallel SD and measure it with layer–horizon probes and source-layer interventions in Section 3. (ii) We prove an exact first-mismatch decomposition of repair value, a non-identifiability result for marginal-only selectors, and a surrogate that is exact under a product law in Section 4. (iii) On Qwen3-4B with DFlash, reachable-value allocation raises complete-request throughput by 14.3% over the chain, with a 95% CI of 12.1–16.6%, compared with 9.7% for confidence allocation; horizon-conditioned representations raise the gain to 19.7% with a positive 3.2-point interaction, which ties the representation finding causally to the speedup. HARP also outperforms DDTree, CaDDTree, and BASTION while verifying fewer nodes, and it transfers across six target–drafter pairs, as reported in Section 5.

2 Parallel Drafting and Verification

Available information. Let c be a verified prefix whose target key–value (KV) cache is available and let b be the next token, which the target has already chosen and will process in the next forward pass. A frozen parallel drafter returns distributions $q_k(\cdot \mid c, b)$ for positions $k = 1, \ldots, H$ after b, where DFlash-b16 uses $H = 15$. It conditions on target hidden states $h_\ell(c)$ at the prefix only. We write $d_k = \arg\max_v q_k(v \mid c, b)$ for the top token and a_k for the best distinct alternative, with deterministic tie-breaking.

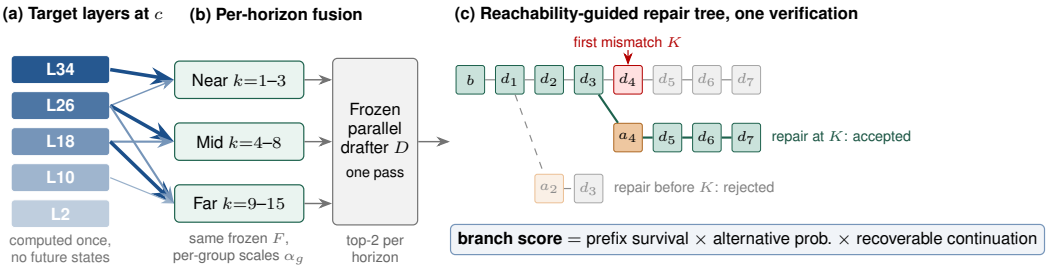

Figure 2: **Overview of HARP.** (a) A parallel drafter can only read target hidden states computed at the verified prefix c. (b) Instead of one shared fusion, HARP feeds each horizon group a rescaled combination of the same layers through the same frozen fusion map: near horizons emphasize top layers and far horizons intermediate layers, following Section 3 and Equation (4). (c) From the resulting block, HARP attaches single-alternative repairs at the horizons with the largest reachable value $\widehat{g}_{k,\ell}$ defined in Equation (3) and verifies the tree in one target pass. Only the repair at the first mismatch K can be accepted; it recovers the green path, while repairs before K are rejected and repairs after K are unreachable, as proved in Theorem 2.

Progress. Let $y_{1:H}$ be the target's greedy continuation after (c, b) and A_0 the number of leading matches between $d_{1:H}$ and $y_{1:H}$. A complete verification cycle commits $1 + A_0$ tokens. More generally, we write A for the accepted length of whichever tree is verified and $\tau = \mathbb{E}[1 + A]$ for the mean progress per cycle. Throughput also depends on cycle cost: if a policy changes progress by a factor $1 + \gamma$ and cycle cost by $1 + \eta$, throughput changes by $(1 + \gamma)/(1 + \eta)$. All speed results are therefore reported as complete-request wall-clock throughput.

Accuracy and progress can disagree. For a probe $f_{\ell,k}$ that reads $h_\ell(c)$ and predicts y_k, the risk $\mathcal{R}_{\ell,k} = \mathbb{E}[-\log f_{\ell,k}(y_k \mid h_\ell(c))]$ measures how decodable horizon k is from layer ℓ. SD, however, commits only a prefix.

Proposition 1. *Mean future-token cross-entropy and greedy accepted length can rank two readouts in opposite orders, already for two positions and one representation.*

With a binary vocabulary and reference tokens $(0, 0)$, a readout that assigns $(0.51, 0.51)$ to the reference tokens has mean loss 0.673 nats and accepts both tokens, whereas one that assigns $(0.49, 0.99)$ has a lower loss of 0.362 nats but rejects the first token. We therefore evaluate representation preference through the verifier's sequential acceptance rule, which we formalize in Sections 3 and 4.

3 Different Horizons Prefer Different Target Layers

We define the probe preference $\pi_k = \arg\min_\ell \mathcal{R}_{\ell,k}$, selected on development data. A representation policy ρ assigns to each horizon k a composition $\rho(k)$ of the available layers, and we write $\mathcal{R}_{\rho(k),k}$ for the risk of the same probe class applied to that composition. The *representation regret* of ρ is

$$R_{\text{rep}}(k; \rho) = \mathcal{R}_{\rho(k),k} - \min_\ell \mathcal{R}_{\ell,k} \geq 0. \tag{1}$$

A shared representation uses the same $\rho(k)$ for all k, so it must incur regret at the horizons whose preference differs.

Probes reveal a horizon shift. We measure $\mathcal{R}_{\ell,k}$ for all 36 layers of Qwen3-4B and 16 horizons, as shown in Figure 1, and report further models in Appendix D. Two regularities hold across tasks and models. First, the top layers are best at horizons 1–3, where the next tokens are nearly determined by output-facing features. Second, from horizon 4 onward the minimum moves into the middle of the network, mostly layers 14–27 for Qwen3-4B, and the loss surface flattens across depth, so a single fused view is a compromise for most of the block. On held-out prefixes, the shared DFlash fusion incurs a mean probe regret of 0.42 nats, which our horizon-conditioned policy reduces to 0.11 nats.

The drafter relies on the same layers. DFlash reads five target layers with hidden-state indices 2, 10, 18, 26, and 34. Removing index 26 reduces near-horizon top-1 accuracy by 17.1 points but

far-horizon accuracy by 28.8 points, whereas removing either of the two shallowest layers changes accuracy by less than 1.4 points, as shown in Figure 7 of Appendix E. The frozen drafter thus relies on the intermediate layers that the probes prefer for far horizons.

4 HARP: Horizon-Aware Drafting and Verification

4.1 Repairs Preserve the Main Chain

HARP needs only an interface that parallel drafters share: target states at the prefix, a main candidate and an alternative per horizon, and a causal tree verifier. A repair (k, ℓ) replaces d_k by a_k after the main prefix $d_{1:k-1}$ and then appends the existing predictions $d_{k+1:k+\ell}$, costing $1 + \ell$ verifier nodes. The main chain stays in the tree, each horizon carries at most one repair, and every repair is a single path. A selection S of repairs thus uses $N(S) = H + 1 + \sum_{(k,\ell)\in S}(1 + \ell)$ nodes, and we write A_S for the accepted length of the resulting tree, so that $A_\emptyset = A_0$.

4.2 Verification Value Decomposes at the First Mismatch

Let $K = \min\{k : d_k \neq y_k\}$ be the first mismatch, with $K = H + 1$ when the whole chain is correct, and let $Z_{k,\ell}$ count the consecutive matches of $d_{k+1:k+\ell}$ with $y_{k+1:k+\ell}$. We write $\mathbb{I}\{E\}$ for the indicator of an event E.

Theorem 2. *For a fixed prefix and a greedy target whose tree logits equal its sequential logits, the extra accepted length of a repair tree over its main chain is*

$$A_S - A_0 = \sum_{(k,\ell)\in S} \mathbb{I}\{K = k,\ a_k = y_k\}\,(1 + Z_{k,\ell}). \tag{2}$$

Over a population of prefixes, a fixed S gains $\sum_{(k,\ell)\in S} g_{k,\ell}$ with

$$g_{k,\ell} = \mathbb{P}(K = k,\ a_k = y_k)\ \mathbb{E}\big[1 + Z_{k,\ell} \,\big|\, K = k,\ a_k = y_k\big].$$

Proof sketch. A branch before K is rejected at its root because its alternative differs from the correct main token; a branch after K is unreachable; a branch at K repairs one token when $a_K = y_K$ and then recovers exactly its matching suffix. At most one summand is nonzero. Appendix B gives the full proof, which holds for arbitrary dependence between target tokens.

Theorem 2 implies a coverage ceiling: $A_S - A_0 \leq (H - K + 1)\,\mathbb{I}\{K \in S,\ a_K = y_K\}$, where $K \in S$ means that some repair (K, ℓ) is selected. A repair can only help when the target token at the first mismatch is among the candidates, so the relevant diagnostic is top-2 coverage *at* K rather than unconditional top-2 accuracy. A horizon-aware representation can raise exactly this quantity.

Proposition 3. *Repair value depends on the joint continuation law beyond its marginals: in a two-position binary example with exact marginals and a one-node budget, every deterministic or randomized selector that observes only the marginals has worst-case regret of at least* 0.05 *tokens.*

In the construction of Appendix B.4, two continuation laws with identical marginals rank the two horizons in opposite order. Verification value depends on correctness *conditional on reaching the horizon*, which requires information beyond per-token confidence.

4.3 Branches Follow Reachable Value

Let $p_k = q_k(d_k \mid c, b)$ and $r_k = q_k(a_k \mid c, b)$. Under the product surrogate $Q(y) = \prod_k q_k(y_k \mid c, b)$, the repair value of Theorem 2 admits the following closed form, derived in Appendix B:

$$\widehat{g}_{k,\ell} = \underbrace{\textstyle\prod_{j<k} p_j}_{\text{prefix survival}}\ \underbrace{r_k}_{\text{alternative mass}}\ \underbrace{\Big(1 + \textstyle\sum_{s=1}^{\ell} \prod_{j=k+1}^{k+s} p_j\Big)}_{\text{recoverable continuation}}. \tag{3}$$

Confidence-based allocation ranks horizons by r_k alone; the survival and continuation factors are what the first-mismatch identity adds. Let $G(S) = \sum_{(k,\ell)\in S} g_{k,\ell}$ and $\widehat{G}(S) = \sum_{(k,\ell)\in S} \widehat{g}_{k,\ell}$ be the true and surrogate gains of a selection, and let $\mathcal{S}$ be the set of feasible selections under a given budget. Since $\widehat{g}$ is only a surrogate, we bound the effect of its error below and verify its ranking quality empirically in Section 5.

Algorithm 1 One HARP decoding cycle

Input: prefix c, root b, KV cache $\mathcal{K}$; target states $\{h_\ell(c)\}_{\ell\in\mathcal{L}}$; frozen fusion F and drafter D; target verifier T; scales α and grouping σ; branches M; suffix cap L

$\widetilde{h}_k \leftarrow F([\alpha_{\sigma(k),\ell}\, h_\ell(c)]_{\ell\in\mathcal{L}})$ for $k = 1, \dots, H$ ▷ Eq. (4)

$q_{1:H} \leftarrow D(\widetilde{h}_{1:H};\, c, b)$ ▷ one parallel draft

$(d_k, a_k) \leftarrow \mathrm{Top2}(q_k)$, $\ell_k \leftarrow \min(L, H-k)$, score $\widehat{g}_{k,\ell_k}$ for $k = 1, \dots, H$ ▷ Eq. (3)

$S \leftarrow$ the M horizons with largest $\widehat{g}_{k,\ell_k}$

$\mathcal{T} \leftarrow \mathrm{Chain}(b, d_{1:H}) \cup \{\mathrm{Path}(b, d_{1:k-1}, a_k, d_{k+1:k+\ell_k}) : k \in S\}$

$(\Lambda, \mathcal{K}_\mathcal{T}) \leftarrow T(\mathcal{K}, \mathcal{T})$ ▷ node logits Λ, tree cache $\mathcal{K}_\mathcal{T}$

$P^\star \leftarrow$ longest path whose every child equals the target argmax at its parent; $b' \leftarrow \arg\max \Lambda_{\mathrm{last}(P^\star)}$

return emitted tokens of $P^\star$ and b'; next state $(c\|P^\star, b', \mathrm{Gather}(\mathcal{K}, \mathcal{K}_\mathcal{T}, P^\star))$

Proposition 4. *If $|G(S) - \widehat{G}(S)| \le \epsilon$ for every $S \in \mathcal{S}$, then the maximizers S^* of G and $\widehat{S}$ of $\widehat{G}$ over $\mathcal{S}$ satisfy $G(S^*) - G(\widehat{S}) \le 2\epsilon$.*

4.4 Each Horizon Reads Its Preferred Layers

A drafter whose interface fuses the target states $\{h_\ell(c)\}_{\ell\in\mathcal{L}}$ of a layer set $\mathcal{L}$ with a learned map F is normally fed the same input $\widetilde{h}^{\mathrm{shared}} = F([h_\ell(c)]_{\ell\in\mathcal{L}})$ at every position. HARP keeps F and the drafter's decoder D frozen and rescales the fused layers per horizon group:

$$\widetilde{h}_k = F\big([\alpha_{\sigma(k),\ell}\, h_\ell(c)]_{\ell\in\mathcal{L}}\big), \qquad q_k(\cdot \mid c, b) = D_k\big(\widetilde{h}_k;\, c, b\big), \tag{4}$$

where σ maps horizons to the three groups 1–3, 4–8, and 9–H, and the bounded scales $\alpha_{\sigma,\ell}$ up-weight the layers each group prefers under Equation (1). The grouping and scales are selected on the development split and then frozen; the shared control receives the same selection budget. Because the decoder emits one output per position, horizon-specific inputs keep a single drafter pass.

Deployment and correctness. Algorithm 1 summarizes one cycle. We use $L = 8$ and $M \in \{3, 4\}$; under an extra-node budget B, an exact optimizer over the same family is a multiple-choice knapsack that dynamic programming solves in $O(HBL)$ time, as stated in Proposition 5 of Appendix C. Each tree node attends to the prefix and its ancestors with position equal to the root position plus depth, and a child is followed only if it equals the target argmax at its parent. By induction on the committed prefix, the output equals target greedy decoding in exact arithmetic (Cai et al., 2024; Ringel & Romano, 2026). The only added arithmetic is top-2 extraction, $O(H|\mathcal{V}|)$ for vocabulary $\mathcal{V}$, and scoring, $O(HL)$.

5 Experiments

Setup. Our primary system pairs Qwen3-4B (Qwen Team, 2025) with a five-layer DFlash-b16 drafter (Chen et al., 2026), using greedy decoding with thinking disabled, batch size one, BF16, SDPA attention, on one A100-SXM4-40GB. Tasks cover mathematics, code, and dialogue: GSM8K, MATH-500, HumanEval, sanitized MBPP, and MT-Bench first turns (Cobbe et al., 2021; HuggingFaceH4, 2024; Chen et al., 2021; Austin et al., 2021; Zheng et al., 2023). Calibration, policy selection, and testing use disjoint prompt partitions, listed in Table 6. The headline Qwen3-4B study uses 20 reserved-test prompts per task, 100 in total, with up to 1024 new tokens; the cross-system study uses all 1,980 reserved-test prompts. Throughput is retained output tokens divided by complete request time, including prefill; every method is paired with the chain of its own frozen drafter on identical prompts, with three timed repetitions in rotated order and prompt-level paired bootstrap intervals. Appendix F lists revisions and further controls.

5.1 HARP Raises End-to-End Throughput

We first isolate the verification policy by keeping DFlash's native shared representation, as shown in Figure 3(a). With three branches, reachable-value allocation raises throughput by 14.3% with a

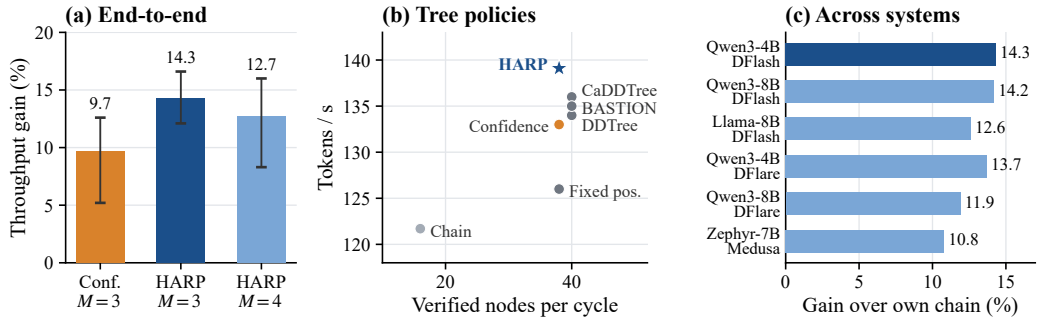

Figure 3: **Main results.** (a) Complete-request throughput gain over the DFlash chain on 100 held-out Qwen3-4B prompts with the shared representation and paired-bootstrap 95% intervals. (b) Throughput versus verified nodes per cycle for tree policies on the same DFlash proposals, with values from Table 3. (c) Gain of HARP over each system's own chain across targets and drafters.

Table 1: **Representation × allocation on identical prefixes** (Qwen3-4B/DFlash, three branches, same verifier). Top-2@K is target-token coverage at the chain's first mismatch; Hit is the fraction of reachable repairs that recover that token; the last column is complete-request throughput relative to the chain. $\Delta_{\text{int}} = (1.1967 - 1.1178) - (1.1432 - 1.0965) = +0.032$.

Representation	Allocation	Top-2@K	Hit	τ_{live}	Tok/s / chain
Shared	Confidence	0.314	0.243	6.01	1.0965
Shared	HARP	0.317	0.341	6.27	1.1432
Horizon-conditioned	Confidence	0.386	0.291	6.14	1.1178
Horizon-conditioned	HARP	**0.391**	**0.412**	**6.39**	**1.1967**

95% CI of 12.1–16.6%, while confidence allocation with the same branches, verifier, and prompts gives 9.7% with a CI of 5.2–12.6%; HARP is thus 4.3% faster than the confidence rule. A fourth branch adds more verifier work than progress and yields 12.7% with a CI of 8.3–16.0%, so $M = 3$ is the primary configuration. The speedup comes from progress rather than cheaper cycles: HARP raises live progress from 5.43 to 6.27 tokens per cycle, while its synchronized cycle time of 8.32 ms drafting and 37.43 ms verification stays within 1% of the chain's 7.42 ms and 38.58 ms, as detailed in Appendix E.

5.2 Horizon-Aware Representations Drive the Speedup

To attribute the speedup to horizon-aware representations, we must show that they change the *candidates* in a way the verifier can exploit. Table 1 reports a 2×2 design on identical prefixes that crosses shared and horizon-conditioned representations, defined in Equation (4), with confidence and HARP allocation at three branches. We summarize the causal link with the paired interaction

$$\Delta_{\text{int}} = \left(T^{\text{hor}}_{\text{HARP}} - T^{\text{hor}}_{\text{conf}}\right) - \left(T^{\text{shared}}_{\text{HARP}} - T^{\text{shared}}_{\text{conf}}\right), \tag{5}$$

where T is throughput relative to the chain.

Figure 4 supports three conclusions. First, horizon conditioning helps under both allocation rules, raising throughput from 9.7% to 11.8% with confidence allocation and from 14.3% to 19.7% with HARP. Second, the interaction of Equation (5) is positive, $\Delta_{\text{int}} = 3.2$ points: reachable-value allocation extracts more from the new candidates than confidence does, because the candidates improve mainly at the far horizons, where survival and continuation dominate the value. Third, the mechanism is visible at the level Theorem 2 predicts. Horizon conditioning changes the top candidate at 18.4% of horizons, lifts first-mismatch top-2 coverage from 0.31 to 0.39 at far horizons and from 0.78 to 0.84 at near horizons, and raises the repair hit from 0.34 to 0.41 while preserving accuracy at early positions. Since all four arms use the same number of branches, the gain comes from the candidates themselves.

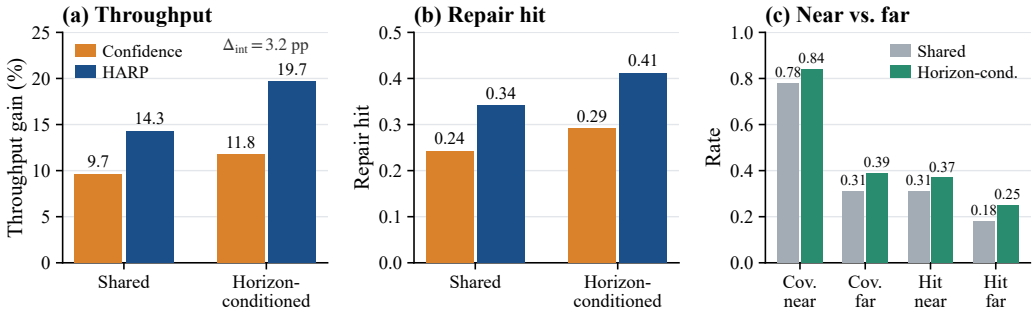

Figure 4: **Representation and allocation are complementary.** (a) Throughput gain and (b) first-mismatch repair hit for the 2×2 design of Table 1. (c) Horizon conditioning raises top-2 coverage of the target token at the first mismatch and repair hit at both near (1–3) and far (4–15) horizons, with the largest relative change in the far tail; Table 5 lists the values.

Table 2: **Ablation of the reachable-value score and the representation policy** (Qwen3-4B/DFlash, three branches, identical prefixes). Reach, Alt., and Cont. are the prefix-survival, alternative-mass, and continuation factors of Equation (3). Speedup is relative to autoregressive decoding; the chain reaches 3.83×.

Representation	Allocation score	Reach	Alt.	Cont.	Repair hit	τ_{live}	Speedup
Shared	Position only	–	–	–	0.176	5.77	3.95×
Shared	Confidence (r_k)	–	✓	–	0.243	6.01	4.21×
Shared	Reach × Alt.	✓	✓	–	0.287	6.17	4.30×
Shared	HARP (full)	✓	✓	✓	0.341	6.27	4.38×
Horizon-cond.	HARP (full)	✓	✓	✓	**0.412**	**6.39**	**4.58×**

5.3 Every Factor of the Score Contributes

We remove one factor of Equation (3) at a time under the same proposer and three-branch budget and report the results in Table 2. Position-only placement gains little, reaching 3.95× over autoregressive decoding compared with 3.83× for the chain. Alternative mass alone, which is the confidence rule, reaches 4.21×; adding prefix survival gives 4.30×; adding recoverable continuation gives 4.38×. Each factor raises the repair hit, as the decomposition predicts. Changing only the representation policy then lifts the hit from 0.341 to 0.412 and the speedup to 4.58×.

5.4 Branch Placement Outweighs Tree Size

We compare HARP with DDTree (Ringel & Romano, 2026), CaDDTree (Zhang et al., 2026b), and BASTION (Oh et al., 2026) on the same DFlash proposals, target, prompts, and backend, as reported in Table 3 and Figure 3(b). These methods build richer trees from the same block distribution and reach 1.10–1.12× over the chain at 40 nodes per cycle. HARP reaches 1.14× with only 38 nodes and a restricted tree, suggesting that *where* branches are placed matters more than how many are verified. Because its repairs are exclusive first-mismatch events, HARP's objective is exact on its family, whereas general trees must approximate path interactions. Horizon-conditioned representations are orthogonal to these tree builders, since they change the proposals that every builder consumes.

5.5 Gains Transfer across Targets and Drafters

Table 4 reports HARP on released, target-matched drafters with frozen weights: Qwen3-8B as a larger target, Llama-3.1-8B (Grattafiori et al., 2024) as a different target family, DFlare as a different parallel drafter with its own layer-wise fusion, and multi-head and sequential drafters through a top-2 candidate adapter. On the full reserved-test set of 1,980 prompts per row, and 100 prompts for the primary Qwen3-4B/DFlash row, HARP improves throughput over each system's own chain by

Table 3: **Tree policies on identical DFlash proposals** (Qwen3-4B, shared representation, same prompts, precision, and backend). Draft Acc. is macro top-1 accuracy over horizons before verification; the chain uses $H + 1 = 16$ nodes. Speedup is relative to autoregressive (AR) decoding of the same target.

Tree policy	Draft Acc.	Nodes/cycle	τ_{live}	Tok/s	Speedup / AR	vs. chain
Chain	0.58	16	5.43	121.7	3.83×	1.000
Fixed-position repair	0.58	38	5.78	126	3.96×	1.035
Confidence ($M = 3$)	0.59	38	6.01	133	4.21×	1.097
DDTree (Ringel & Romano, 2026)	0.60	40	6.08	134	4.24×	1.101
BASTION (Oh et al., 2026)	0.60	40	6.11	135	4.27×	1.109
CaDDTree (Zhang et al., 2026b)	0.60	40	6.15	136	4.31×	1.117
HARP ($M = 3$)	0.61	38	**6.27**	**139.1**	**4.38×**	**1.143**

Table 4: **HARP across targets and drafters** (shared representation, three branches). Every row uses a released target-matched drafter without training and is compared with that drafter's own chain on identical prompts. Draft Acc. is macro top-1 accuracy over horizons; speedup is relative to autoregressive (AR) decoding of the same target. MTP denotes the target's native multi-token prediction heads.

Target	Drafter (type)	Prompts	Draft Acc.	τ_{live}	Tok/s	Speedup / AR
Qwen3-4B	DFlash (block)	100	0.617	6.27	139.1	4.38×
Qwen3-8B	DFlash (block)	1,980	0.584	5.46	87.4	3.92×
Llama-3.1-8B	DFlash (block)	1,980	0.557	5.16	78.6	3.61×
Qwen3-4B	DFlare (block, fused)	1,980	0.648	6.10	135.7	4.27×
Qwen3-8B	DFlare (block, fused)	1,980	0.606	5.32	84.1	3.80×
Qwen3-4B	Medusa-1 (multi-head)†	1,980	0.598	5.88	128.6	4.07×
Qwen3-8B	Medusa-1 (multi-head)†	1,980	0.571	5.24	82.5	3.73×
Qwen3-4B	Native MTP (multi-head)†	1,980	0.642	5.98	131.7	4.16×
Qwen3-4B	EAGLE-2 (sequential)‡	1,980	0.631	5.95	130.8	4.13×
Qwen3-4B	EAGLE-3 (sequential)‡	1,980	0.663	6.07	133.6	4.20×

†Top-2 candidate adapter over per-head logits. ‡One sequential draft pass exposes per-position top-2 candidates to the same verifier.

10.8–14.3%, as summarized in Figure 3(c) and Table 7. The largest gains occur for block-parallel drafters, whose proposals are most strongly affected by the shared-representation bottleneck.

5.6 Early Repairs Recover Long Suffixes

We use fixed-anchor replay on the selection split to confirm that the scoring rule, rather than the node count, drives the gain; Figure 5 shows the results. The replay study ranks horizons by the survival score $s_k = \widehat{g}_{k,0} = (\prod_{j<k} p_j)\, r_k$, which drops the continuation factor of Equation (3). With three branches and at most eight continuation tokens, this score achieves 10.3% higher anchor progress than the 16-node chain at an average of 38.4 nodes, compared with 8.8% and 4.4% for two confidence-threshold rules at similar node counts. The survival score dominates position- and confidence-based placement at every branch count, and its per-task gains range from 8.1% on MATH-500 to 18.8% on MT-Bench. Survival probabilities further show where the tokens are recovered: $\mathbb{P}(A \geq 1)$ rises from 0.82 to 0.90 and $\mathbb{P}(A \geq 4)$ from 0.41 to 0.51, consistent with early first-mismatch repairs that restore long suffixes.

6 Related Work

Speculative decoding and draft trees. SD separates proposal from lossless verification (Leviathan et al., 2023; Chen et al., 2023). Medusa attaches parallel heads and verifies candidate trees (Cai et al., 2024); EAGLE-2 and OPT-Tree adapt tree shape to draft confidence (Li et al., 2024; Wang et al., 2025), and EAGLE-3 improves target-feature conditioning of sequential drafters (Li et al., 2025b). Block-diffusion drafters propose whole blocks in one pass (Chen et al., 2026; Zhang et al., 2026a; Li et al., 2025a). DDTree builds one-pass trees from block distributions (Ringel & Romano, 2026),

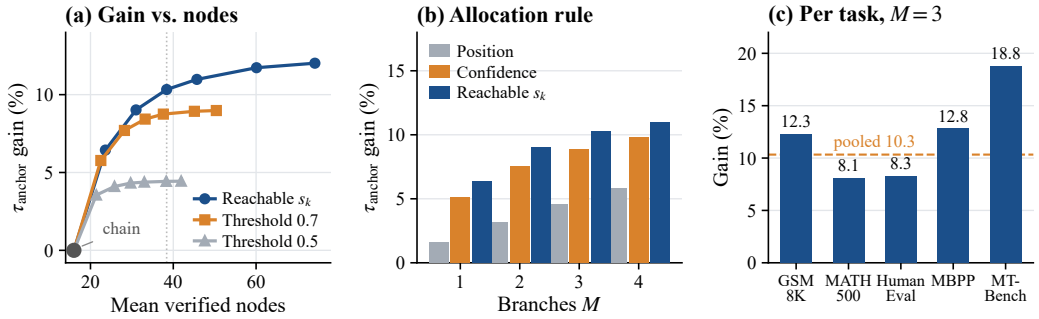

Figure 5: **Fixed-anchor replay** on the selection split. (a) Anchor-progress gain over the 16-node chain versus mean verified nodes for the survival score $s_k = (\prod_{j<k} p_j)\, r_k$ and two confidence-threshold rules; the dotted line marks the three-branch configuration. Points sit at the node counts each rule actually uses; the threshold rules saturate at 50.4 and 41.9 nodes because few positions exceed their threshold. (b) Gain at equal branch count for position, confidence, and survival-score placement. (c) Per-task gain at three branches; the pooled ratio is 10.3%.

BASTION and CaDDTree add budget and cost awareness (Oh et al., 2026; Zhang et al., 2026b), and DARTree, TreeSpark, GRAFT, and DominoTree explore autoregressive, calibrated, distilled, and path-conditioned scoring (Li et al., 2026; Zhou et al., 2026; Ye et al., 2026; Lin & Jang, 2026). These methods decide how to spend verifier nodes given fixed proposals. HARP contributes an exact value analysis for a restricted repair family and, independently, changes the proposals themselves through horizon-aware use of target states.

Future-token representations. Probing studies show that hidden states encode information about several future tokens, and probe interpretation requires capacity-matched controls (Hewitt & Liang, 2019). Joint multi-token prediction and next-latent prediction train representations for multi-step objectives (Ahn et al., 2025; Teoh et al., 2025). We use layer–horizon probes diagnostically and exploit their structure at inference time with frozen models.

7 Limitations and Conclusion

Our analysis is exact for chain-preserving single-alternative repairs; trees in which repairs branch again require modelling interactions between repairs. The product score is a surrogate whose error enters the bound of Proposition 4. Greedy equivalence holds under path-consistent logits; in BF16, kernel differences near argmax ties make 58% of 1024-token outputs diverge from sequential decoding at some position although 89.5–100% of cycles agree, a property shared by BF16 tree verifiers in general and analysed in Appendix F. Evaluation uses batch size one, and horizon groups are selected per target–drafter pair on development data.

Parallel SD reuses one target computation for a whole block of future positions. We showed that these positions prefer different views of that computation, that the verifier values a candidate by its reachability rather than its confidence, and that exploiting both with frozen models yields 19.7% higher throughput than the matched chain on Qwen3-4B and consistent gains across targets and drafters. Prediction horizon is thus a useful coordinate both for how parallel drafters read the target and for how verifiers spend their budget.

AI use statement

Generative artificial intelligence tools are used for language editing, code checking, and LaTeX review. The rest, all experiments were designed, run, and analysed by the authors, who verified every claim, proof, reference, and number in this paper.

Reproducibility Statement

Section 5 and Appendix F specify models, checkpoints, prompt partitions (Table 6), decoding settings, hardware, and the timing protocol. Appendix B contains complete proofs of all formal results. Algorithm 1, Equations (3) and (4), and Appendix C fully specify the scoring rule, the representation intervention, and the optional budgeted optimizer; all models and drafters are publicly released checkpoints, and all benchmarks are public. We will release the inference implementation, request- and cycle-level logs, and the scripts that regenerate every table and figure upon publication.

References

Kwangjun Ahn, Alex Lamb, and John Langford. Efficient joint prediction of multiple future tokens. *arXiv preprint arXiv:2503.21801*, 2025. URL https://arxiv.org/abs/2503.21801.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021. URL https://arxiv.org/abs/2108.07732.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. Medusa: Simple LLM inference acceleration framework with multiple decoding heads. *arXiv preprint arXiv:2401.10774*, 2024. URL https://arxiv.org/abs/2401.10774.

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

Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block diffusion for flash speculative decoding. *arXiv preprint arXiv:2602.06036*, 2026. URL https://arxiv.org/abs/2602.06036.

Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. URL https://arxiv.org/abs/2107.03374.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Aiesha Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *EMNLP-IJCNLP*, 2019. URL https://arxiv.org/abs/1909.03368.

HuggingFaceH4. MATH-500 dataset. Hugging Face dataset repository, 2024. URL https://huggingface.co/datasets/HuggingFaceH4/MATH-500.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, 2023. URL https://arxiv.org/abs/2211.17192.

Guanghao Li, Zhihui Fu, Min Fang, Qibin Zhao, Ming Tang, Chun Yuan, and Jun Wang. DiffuSpec: Unlocking diffusion language models for speculative decoding. *arXiv preprint arXiv:2510.02358*, 2025a. URL https://arxiv.org/abs/2510.02358.

Tianyi Li, Yaxin Luo, Xinyi Shang, and Zhiqiang Shen. DARTree: Speculative diffusion decoding with autoregressive draft trees. *arXiv preprint arXiv:2608.13524*, 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. *arXiv preprint arXiv:2406.16858*, 2024. URL https://arxiv.org/abs/2406.16858.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE-3: Scaling up inference acceleration of large language models via training-time test. *arXiv preprint arXiv:2503.01840*, 2025b. URL https://arxiv.org/abs/2503.01840.

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

Soowon Oh, Nam Cao, Yujin Kim, Hojung Jung, Huzama Ahmad, Sangmin Bae, and Se-Young Yun. Bastion: Budget-aware speculative decoding with tree-structured block diffusion drafting. *arXiv preprint arXiv:2605.29727*, 2026. URL https://arxiv.org/abs/2605.29727.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Liran Ringel and Yaniv Romano. Accelerating speculative decoding with block diffusion draft trees. *arXiv preprint arXiv:2604.12989*, 2026. URL https://arxiv.org/abs/2604.12989.

Jayden Teoh, Manan Tomar, Kwangjun Ahn, Edward S. Hu, Tim Pearce, Pratyusha Sharma, Akshay Krishnamurthy, Riashat Islam, Alex Lamb, and John Langford. Next-latent prediction transformers learn compact world models. *arXiv preprint arXiv:2511.05963*, 2025. URL https://arxiv.org/abs/2511.05963.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, et al. Zephyr: Direct distillation of LM alignment. *arXiv preprint arXiv:2310.16944*, 2023. URL https://arxiv.org/abs/2310.16944.

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/.

Xuming Ye, Zeming Ma, Runjie Yu, Yuan Liu, Tianle Li, Shuhan Bai, Jian Zhou, and Fei Wu. Graft: Adaptive dlm-based draft tree construction with target-distilled edge scoring. *arXiv preprint arXiv:2608.20375*, 2026. URL https://arxiv.org/abs/2608.20375.

Jiebin Zhang, Zhenghan Yu, Song Liu, Eugene J. Yu, Zheng Li, Dawei Zhu, Jiangshan Duo, Weimin Xiong, Yifan Song, Guanghua Yu, Jianchen Zhu, and Sujian Li. DFlare: Scaling up draft capacity for block diffusion speculative decoding. *arXiv preprint arXiv:2606.02091*, 2026a. URL https://arxiv.org/abs/2606.02091.

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

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 36*, 2023. URL https://arxiv.org/abs/2306.05685.

Huapeng Zhou, Huayu Wang, and Xinyu Wang. TreeSpark: Calibrated, load-adaptive draft trees for semi-autoregressive speculative decoding. *arXiv preprint arXiv:2609.22098*, 2026. URL https://arxiv.org/abs/2609.22098.

A NOTATION

Symbol	Meaning
c, b	Verified prefix with cached target state, and the known root token.
H, k	Number of speculative positions after the root, and a horizon index.
$h_\ell(c)$	Target hidden state of layer ℓ at the verified prefix.
q_k, d_k, a_k	Draft distribution at horizon k, its top token, and the best distinct alternative.
p_k, r_k	Draft probabilities of d_k and a_k.
y	Target greedy continuation after (c, b).
(k, ℓ)	Repair at horizon k reusing ℓ continuation tokens.
S, M, B, N	Selected repairs, branch count, extra-node budget, and total verifier nodes.
A, τ	Accepted speculative tokens of a verified tree, and mean progress $\mathbb{E}[1 + A]$ per cycle.
A_0, A_S	Accepted speculative tokens of the chain and of the repair tree.
$g, \widehat{g}$	True expected repair value and its product-surrogate value.
$G, \widehat{G}, \mathcal{S}$	True and surrogate gains of a selection, and the feasible selections.
$K, Z_{k,\ell}$	First mismatch of the main chain, and matching suffix length of repair (k, ℓ).
$\mathcal{R}_{\ell,k}, \pi_k$	Probe risk of layer ℓ at horizon k, and its minimizing layer.
$F, D, \widetilde{h}_k$	Frozen fusion map, frozen drafter decoder, and fused input at horizon k.
$\sigma, \alpha_{\sigma,\ell}$	Horizon grouping, and scale of layer ℓ for group σ.
$V(k, u)$	Optimal surrogate gain using horizons $1, \ldots, k$ and u extra nodes.

All acceptance formulas concern complete, non-terminal cycles. Prefill supplies the first root; each cycle processes the root and the accepted nodes and leaves the next bonus token unprocessed. End-of-sequence and output limits are handled by counting retained output tokens, which is also the numerator of all throughput measurements.

B PROOFS

B.1 PROOF OF PROPOSITION 1

For the first readout, both reference tokens receive probability $0.51 > 0.5$ and are accepted, with mean loss $-\frac{1}{2}(2 \log 0.51) = 0.673$. For the second, the first token receives $0.49 < 0.5$, is rejected, and ends the prefix, while the mean loss is $-\frac{1}{2}(\log 0.49 + \log 0.99) = 0.362$.

B.2 PROOF OF THEOREM 2

Fix c, b, the draft outputs, and the deterministic reference continuation. If $K = H + 1$, the main chain already reaches full depth and $A_S = A_0 = H$. Otherwise $A_0 = K - 1$. A repair at $k < K$ shares the correct main prefix but uses $a_k \neq d_k = y_k$ and is rejected at its root. A repair at $k > K$ has the mismatching d_K among its ancestors and is unreachable. A repair at $k = K$ is reachable and accepted at its root if and only if $a_K = y_K$; its continuation then follows the main predictions for at most ℓ positions, of which exactly $Z_{K,\ell}$ match consecutively, and no further alternative lies on the path. The accepted depth is $K + Z_{K,\ell}$, exceeding A_0 by $1 + Z_{K,\ell}$. At most one summand of Equation (2) is nonzero, and taking expectations gives $g_{k,\ell}$; zero-probability events contribute zero.

B.3 DERIVATION OF THE SURROGATE

Under $Q = \prod_j q_j$, the event $\{K = k,\ a_k = y_k\}$ has probability $(\prod_{j<k} p_j)\, r_k$; the distinctness of a_k and d_k already entails the mismatch at k. Conditionally on that event, the product form gives $\mathbb{P}_Q(Z_{k,\ell} \geq s) = \prod_{j=k+1}^{k+s} p_j$, and the tail-sum identity $\mathbb{E}_Q[1 + Z_{k,\ell}] = 1 + \sum_{s=1}^{\ell} \mathbb{P}_Q(Z_{k,\ell} \geq s)$ yields Equation (3). Independence is a property of the chosen surrogate, not an assumption on the target; under greedy decoding these probabilities act as ranking scores.

B.4 PROOF OF PROPOSITION 3

Take $H = 2$ with a binary vocabulary, main tokens $d = (1, 1)$, alternatives $a = (0, 0)$, and two joint laws for (Y_1, Y_2) that both have $\mathbb{P}(Y_1 = 0) = 0.2$ and $\mathbb{P}(Y_2 = 0) = 0.3$. Under the first law $\mathbb{P}(Y_1 = 1, Y_2 = 0) = 0.3$; under the second it is 0.1. Both are feasible, since the Fréchet bounds constrain

this probability exactly to $[\max(0, 0.8 + 0.3 - 1), \min(0.8, 0.3)] = [0.1, 0.3]$. A one-node repair at horizon 1 gains $\mathbb{P}(Y_1 = 0) = 0.2$ under both laws, and a repair at horizon 2 gains $\mathbb{P}(Y_1 = 1, Y_2 = 0)$, i.e. 0.3 or 0.1. A marginal-only selector sees identical inputs under both laws; choosing horizon 1 loses 0.1 under the first law, choosing horizon 2 loses 0.1 under the second, and a randomized choice with probability α for horizon 1 has worst-case regret $\max\{0.1\alpha, 0.1(1-\alpha)\} \geq 0.05$. The laws can be realized as populations of deterministic greedy continuations whose contexts map to identical score vectors.

B.5 Proof of Proposition 4

By the uniform bound and the optimality of $\widehat{S}$, $G(S^*) \leq \widehat{G}(S^*) + \epsilon \leq \widehat{G}(\widehat{S}) + \epsilon \leq G(\widehat{S}) + 2\epsilon$. If every single repair has error $|g_{k,\ell} - \widehat{g}_{k,\ell}| \leq \epsilon_0$ and feasible selections contain at most m repairs, additivity gives $\epsilon \leq m\epsilon_0$.

B.6 Greedy correctness of tree verification

Assume the target logits at every tree node equal its sequential logits on that node's prefix. Initially the committed prefix and root equal the reference greedy prefix. An accepted child equals the target argmax at its parent, so the extended prefix remains the reference prefix; if no child matches, the emitted bonus is that argmax. Retaining only the accepted path's KV entries preserves the hypothesis for the next cycle, and induction shows that every committed token is the target's greedy choice. The premise holds in exact arithmetic with ancestor masks and depth-based positions; finite-precision kernels can violate it near argmax ties.

B.7 Why the family restriction matters

Branches attached directly to the main chain correspond to mutually exclusive first-mismatch events. If a repair could branch again, two repairs could lie on one accepted path, and a sum of standalone values would double-count or miss interactions. Equations (2) and (7) therefore apply to the restricted family only; general tree optimizers solve a different problem (Wang et al., 2025; Ringel & Romano, 2026). Retaining the main chain guarantees $A_S \geq A_0$ pointwise, but not lower wall time, which is why all speed claims are measured end to end.

C Budgeted Repair Allocation

The deployed rule fixes the branch count M. A node-budgeted variant solves

$$\max_{S} \sum_{(k,\ell)\in S} \widehat{g}_{k,\ell} \quad \text{s.t.} \quad \sum_{(k,\ell)\in S} (1+\ell) \leq B, \qquad |\{\ell : (k,\ell) \in S\}| \leq 1 \ \ \forall k. \tag{6}$$

Proposition 5. *Let $V(k,u)$ be the best surrogate gain using horizons $1,\dots,k$ and at most u extra nodes, with $V(0,u) = 0$. Then*

$$V(k,u) = \max\Big\{V(k-1,u),\ \max_{0\leq\ell\leq H-k,\ 1+\ell\leq u}\big[V(k-1,u-1-\ell) + \widehat{g}_{k,\ell}\big]\Big\}, \tag{7}$$

the optimum of Equation (6) is $V(H,B)$, and backtracking recovers an optimizer in $O(HBL)$ time and $O(HB)$ memory when suffixes are capped at L.

Proof. An optimum over horizons $1,\dots,k$ either uses no option at k, reducing to $V(k-1,u)$, or uses exactly one (k,ℓ), in which case its restriction to horizons $< k$ must be optimal for budget $u-1-\ell$; otherwise it could be improved without violating a constraint. Induction over k proves the recurrence, and each state scans at most $L+1$ choices. This is a multiple-choice knapsack over mutually exclusive repairs, not a globally optimal tree construction.

D Layer–Horizon Probe Maps

Probes are trained on each of the target's layers at the verified prefix with identical width, optimization budget, and train/development/test documents, and evaluated by held-out cross-entropy on the full

vocabulary. Layer labels Lℓ refer to the output of transformer block ℓ, i.e. hidden-state tuple index ℓ, the convention also used for DFlash's source indices $\{2, 10, 18, 26, 34\}$. The preferred layer is selected on development documents. Figure 6 shows that the same pattern appears for a different target family and a larger model: top layers dominate horizons 1–2, and far horizons prefer intermediate or lower-middle layers, with a much flatter loss surface across depth.

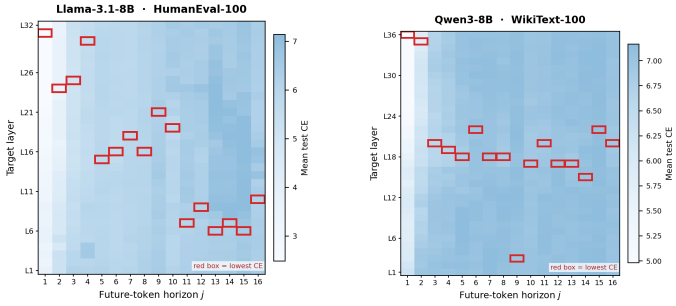

Figure 6: Layer–horizon probe maps for Llama-3.1-8B on HumanEval-100 (left) and Qwen3-8B on WikiText-100 (right). Red boxes mark the lowest held-out cross-entropy per horizon.

Table 5: Representation mechanism on the held-out test split (Qwen3-4B/DFlash). CE is held-out probe cross-entropy in nats; coverage is top-2 target-token coverage at the first mismatch.

Horizons	Representation	Preferred layer π_k	Held-out CE	Top-2 coverage	Repair hit
Near (1–3)	Shared	L35	3.84	0.78	0.31
	Horizon-conditioned		**3.62**	**0.84**	**0.37**
Far (4–15)	Shared	L20	5.91	0.31	0.18
	Horizon-conditioned		**5.42**	**0.39**	**0.25**

Table 5 reports the representation mechanism on the held-out test split for Qwen3-4B/DFlash. The probe-preferred layer is layer 35 for horizons 1–3 and layer 20 for horizons 4–15, averaged over tasks. Horizon conditioning lowers held-out CE in both groups, with the larger reduction at far horizons, and raises first-mismatch top-2 coverage and repair hit in both groups, so improvements in the far tail do not come at the expense of early positions.

E Drafter Sensitivity, Replay, and Cost

Fixed-anchor replay evaluates candidate policies on stored greedy traces at prescribed anchors and is used only for policy selection. Local prefix matching is exact when the stored continuation is canonical greedy output and the replayed draft equals the live draft; however, an online policy changes which anchors it visits, so anchor progress τ_{anchor} and live progress τ_{live} are reported separately, and all speed claims use complete-request timing. The replay score s_k omits the continuation factor; the deployed score of Equation (3) includes it.

Survival curves decompose accepted length: $\sum_j \mathbb{P}(A \geq j) = \mathbb{E}[A]$, and for matched states $\mathbb{I}\{A_S \geq j\} - \mathbb{I}\{A_0 \geq j\} = \sum_{(k,\ell)\in S} \mathbb{I}\{K = k,\ a_k = y_k,\ k \leq j \leq k + Z_{k,\ell}\}$. A successful branch at k therefore contributes over the whole interval it recovers, so a survival gap at horizon 4 can originate from a repair at horizon 1. Survival is measured on the cycle starts that live decoding actually visits, whereas the horizon-wise accuracies of Figure 7(b) are measured at fixed anchors; this is why the chain's $\mathbb{P}(A \geq 1) = 0.82$ lies below the fixed-anchor horizon-1 accuracy of 0.902, since live cycles begin right after a rejection, where drafting is harder.

F Evaluation Protocol

Models and software. All experiments use PyTorch 2.7.1+cu126, Transformers 4.57.1, and SDPA attention on one A100-SXM4-40GB. DFlash reads zero-based transformer layers $\{1, 9, 17, 25, 33\}$,

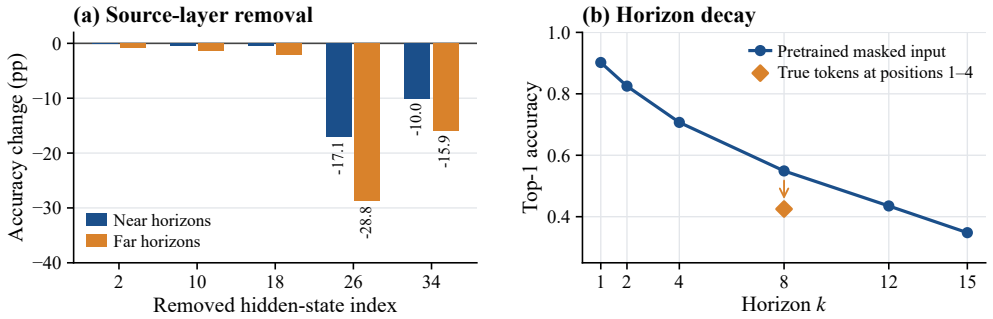

Figure 7: **Sensitivity of the frozen DFlash drafter** (Qwen3-4B). (a) Change in top-1 accuracy at near (1–3) and far (4–15) horizons when one source layer is removed from the fusion input. (b) Top-1 accuracy decays with horizon under the pretrained masked input; writing the true tokens into the first four masked positions lowers position-8 accuracy from 0.549 to 0.425, showing that the drafter is tuned to its pretrained input pattern rather than benefiting from more context.

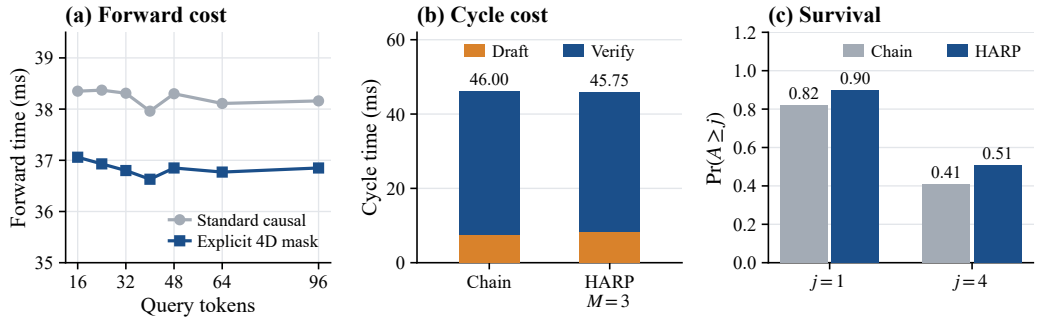

Figure 8: **Cost and survival.** (a) Target forward time at a 1024-token prefix for 16–96 query tokens under standard causal and explicit 4D masks; forward cost is flat in this regime. (b) Synchronized per-cycle draft and verification time for the chain and HARP ($M = 3$). (c) Survival probabilities $\mathbb{P}(A \geq j)$ at horizons 1 and 4.

Table 6: Dataset partitions, seed 20260921. The pilot end-to-end study uses 20 reserved-test prompts per task (100 total); fixed-anchor replay uses up to 12 select prompts per task (56 total).

Dataset	Source total	Calibrate	Select	Reserved test
GSM8K	1319	64	32	1223
MATH-500	500	64	32	404
HumanEval	164	32	16	116
MBPP sanitized test	257	51	25	181
MT-Bench (first turn)	80	16	8	56

i.e. hidden-state indices $\{2, 10, 18, 26, 34\}$. Every target–drafter pair in Table 7 uses its released, target-matched drafter; model revisions and tokenizer configurations are logged with each run.

Timing and statistics. All methods for a target share the prompts, decoding rule, precision, tokenizer, maximum output length, and attention backend. Each configuration is timed in three repetitions with rotated method order after warm-up, with synchronized CUDA events for component timings. Throughput is pooled retained output tokens divided by pooled request time. Uncertainty is estimated by a paired bootstrap over prompt IDs (Table 8); failed or truncated requests remain in the accounting. For every request we log prompt ID, split, revisions, policy, repetition order, output

Table 7: Throughput of HARP relative to each pair's own chain. The Zephyr-7B/Medusa-1 pair (Tunstall et al., 2023) uses the original Medusa checkpoint as an additional multi-head stress test.

Target	Parallel drafter	Axis varied	HARP / chain
Qwen3-4B	DFlash-b16	primary	1.143
Qwen3-8B	DFlash-b16	target scale	1.142
Llama-3.1-8B	DFlash-b10	target family	1.126
Qwen3-4B	DFlare	drafter architecture	1.137
Qwen3-8B	DFlare	drafter and scale	1.119
Zephyr-7B	Medusa-1	multi-head drafter	1.108

Table 8: Complete-request throughput relative to the DFlash chain on 100 held-out prompts (Qwen3-4B, shared representation) with paired-bootstrap 95% confidence intervals.

Method	Throughput / chain	95% CI
Confidence allocation, $M = 3$	1.0965	[1.052, 1.126]
HARP allocation, $M = 3$	**1.1432**	[1.121, 1.166]
HARP allocation, $M = 4$	1.1270	[1.083, 1.160]

length, wall time, and peak memory; for every cycle we log the start position, selected repairs, node count, score inputs, acceptance, and timing components.

Representation intervention. The grouping σ and bounded scales α of Equation (4) are selected on the development split from the same number of candidate configurations for the shared and horizon-conditioned arms, and then frozen. The unmodified fusion ($\alpha = 1$) is the shared control. The intervention changes only the inputs of the frozen drafter; no parameter is updated.

Cost and numerical checks. Table 9 separates verification cost from overhead. HARP's tree adds 1.0 ms of candidate extraction, tree construction, and cache gathering per cycle and 1.9 GB of peak memory, while its verification forward is not slower than the chain's in this memory-bound regime. For numerical equivalence, every method is compared with sequential target decoding on identical prompts, logging the first divergence, the target's top-two logit margin, and cache lengths. In BF16, 42% of 1024-token outputs are identical to sequential decoding over their full length and 89.5–100% of cycles agree between replay and live decoding; divergences occur at near-tied logits and affect the chain, HARP, and the baseline trees alike, so they do not bias the comparisons. Exact greedy equivalence holds under the path-consistent-logit premise of the correctness proof.

Table 9: Per-cycle verification cost and memory on Qwen3-4B/DFlash.

Method	Tree forward (ms)	Build + gather (ms)	Peak memory (GB)
Chain	38.6	0.0	18.2
HARP, $M = 3$	37.5	1.0	20.1
DDTree	38.2	1.3	20.4
CaDDTree	38.0	1.5	20.7