BRANCH-JACOBI: PREFIX-CONDITIONED MULTI-PASS DRAFTING FOR SPECULATIVE DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Block-diffusion drafters predict token blocks in parallel, but later predictions cannot condition on tokens selected earlier in the same pass. We introduce Branch-Jacobi, which reuses one DFlash-style drafter for multi-pass prefix-conditioned refinement. Each pass packs prefix-MASK branches into one forward call. Consecutive first-MASK checks identify the proposal's common prefix with a drafter-autoregressive path; selecting one complete branch extends certification before linear target verification. We prove sound prefix certification for deterministic greedy drafting under fixed cycle conditions and equivalent branch evaluation. Moving-anchor training and causal prefix supervision align the drafter with variable-prefix queries. Across seven math, code, and chat benchmarks on Qwen3-4B and Qwen3-8B at temperatures 0 and 1, a three-layer drafter using three passes per cycle achieves 4.93–$6.40\times$ geometric-mean decode-stage speedups over target autoregressive decoding, exceeding DFlash by 25.3–29.2% while increasing average acceptance length.

1 INTRODUCTION

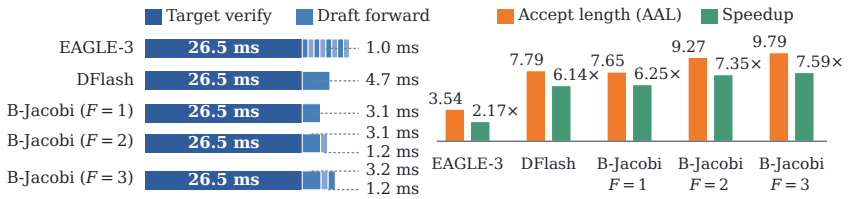

Figure 1: Qwen3-4B/MATH-500 at $T = 0$. Left: cycle-weighted CUDA model-forward costs, excluding each response's first cycle. Right: response-mean AAL and decode-stage speedup from separate uninstrumented runs. EAGLE-3 uses tree size 60.

Autoregressive large language models generate one token at a time, making decoding latency depend on a long sequence of target-model calls. Speculative decoding replaces some of these calls with inexpensive drafting and parallel target verification: a draft model proposes several future tokens, and the target checks them in parallel before committing an accepted prefix (Leviathan et al., 2023; Chen et al., 2023). Its speedup depends on both the number of tokens advanced per verification and the cost of each draft–verify cycle. Autoregressive drafters such as EAGLE-3 model dependencies among draft tokens but retain a sequential draft rollout (Li et al., 2025). DFlash instead predicts an entire block in one forward pass with a lightweight block-diffusion drafter conditioned on target-model features (Chen et al., 2026). Branch-Jacobi builds on this parallel design and addresses its missing within-block conditioning.

Figure 1 examines whether acceptance gains compensate for additional drafting cost, while the target verifies one linear block. Refinement increases acceptance length and decoding speed in this setting, although the fifth pass adds little acceptance and increases TPOT. Table 1 evaluates this trade-off across seven benchmarks.

DFlash's masked positions interact through bidirectional block attention, yet the prediction at position i cannot condition on the realized tokens selected at positions $1, \ldots, i-1$ in the same pass.

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.

Because target verification proceeds causally, an early mismatch discards all later candidates. A recent analysis decomposes this rejection into an information floor caused by unavailable within-block predecessors and a model gap under information already available to the drafter (Qiang et al., 2026). This decomposition motivates conditional redrafting that exposes realized prefixes while retaining parallel block computation.

Figure 2 examines where refinement improves acceptance and how much of the gain from serial conditional drafting it recovers. In the upper panel, $q_i = \Pr(a \geq i \mid a \geq i-1)$, where a counts target-accepted draft tokens; $q_i - q_1$ measures acceptance relative to the first position. The lower panel quantifies overall AAL improvement through gap recovery, $(\mathrm{AAL}_F - \mathrm{AAL}_1)/(\mathrm{AAL}_{\text{Draft-AR}} - \mathrm{AAL}_1)$. Four and five passes recover 99.2% and 100.8% of the serial causal Draft-AR gap. The serial causal Draft-AR control provides a measured AAL reference, while the full-MASK path D defines drafter-side certification. Existing approaches recover within-block information through causal correction modules (Huang et al., 2026b; Cheng et al., 2026; Wang et al., 2026a; Inco AI, 2026) or target-verified candidate trees (Ringel & Romano, 2026; Zhang et al., 2026b). Branch-Jacobi recomputes suffixes after observing proposed prefixes, keeps branching on the draft side, and retains one linear target-verification interface.

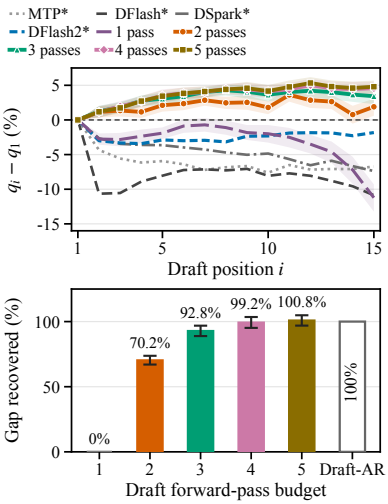

Figure 2: Acceptance trends (top) and gap recovery to the serial causal Draft-AR control (bottom). Our curves use Qwen3-4B/MATH-500 at $T = 0$, with 95% response-bootstrap intervals. All starred curves are adapted from Figure 5 of Inco AI (2026) using Qwen3.5-4B and different sampling settings; they illustrate qualitative position-wise trends (Appendix D).

Branch-Jacobi begins with a DFlash-style block generated from one root token and masked future slots. Each refinement pass constructs one branch for every live prefix length: a branch retains the corresponding prefix of the current proposal and masks the remaining positions. A block-diagonal attention mask isolates the branches, which share target-context keys and values and are packed into one forward call of the same drafter. Consecutive matches between the branches' first-MASK predictions and the current proposal identify its common prefix with the drafter-autoregressive path. At the first mismatch, we select the complete branch anchored at this common prefix and certify its next predicted token; the remaining suffix stays provisional. The same packed forward supplies both consistency checks and complete suffix predictions, so the selected suffix can be used immediately. After a fixed number of passes, the target verifies one linear proposal. This design exchanges additional draft forward passes for stronger path consistency without adding a second drafter, a causal refiner, or target tree nodes.

We train the drafter on moving anchors: for a block with P proposal slots, training exposes an approximately balanced range of ground-truth prefix lengths $k \in \{0, \ldots, P-1\}$ and supervises the masked suffix. A causal prefix objective separately trains generation of the revealed prefix, while the main block computation retains DFlash's bidirectional attention. With deterministic greedy drafter decisions, first-MASK consistency checks certify tokens on the autoregressive path induced by the same drafter; target verification determines which speculative tokens are committed. Across Qwen3-4B and Qwen3-8B at temperatures 0 and 1, three passes achieve the highest measured aggregate speedup among the tested budgets, while five passes yield the highest aggregate acceptance length.

Our contributions are threefold. First, we introduce packed prefix-conditioned refinement that reuses one DFlash-style drafter and retains one linear target verification. Second, we align this inference procedure with moving-anchor suffix training and causal prefix supervision, and characterize the current proposal's common prefix with the drafter-autoregressive path and prove sound certification of an extension from one complete branch. Third, experiments show gains in acceptance length and decoding speed and characterize the trade-off across refinement budgets.

2 Related Work

2.1 Speculative decoding and parallel block drafting

Blockwise parallel decoding predicts multiple positions and retains a verified prefix (Stern et al., 2018). Classical speculative decoding uses an inexpensive draft distribution followed by parallel target-model verification. With the appropriate acceptance and residual-sampling rule, the procedure preserves the target distribution (Leviathan et al., 2023; Chen et al., 2023). EAGLE-3 improves drafter quality by predicting tokens from fused multi-layer target features while following a sequential draft path (Li et al., 2025). Speculative Diffusion Decoding uses a diffusion drafter to propose blocks for target verification (Christopher et al., 2025). DFlash removes most draft-side serialization by predicting all masked positions in one pass from target context features (Chen et al., 2026). Branch-Jacobi retains the DFlash backbone and verifier while adding prefix-conditioned draft forward passes.

2.2 Recovering within-block path information

Parallel position-wise predictions cannot directly depend on the tokens selected earlier in the same sampled path. TreeFlash adds a previous-token-conditioned autoregressive approximation to a one-shot backbone (Rheinboldt et al., 2026). Domino couples DFlash-like parallel features with a GRU causal encoder and low-rank logit residual (Huang et al., 2026b); DSpark's default Markov head uses only the immediately preceding token, while its RNN alternative carries longer history (Cheng et al., 2026). DeLS-Spec combines a fixed long-context DFlash expert with an independently trained short-context causal expert (Zheng & Li, 2026). The DFlash 2 technical release selects a path among each slot's top candidates using adjacent-token scores and also introduces short dynamic convolutions into the backbone (Inco AI, 2026). LiLiCorr scores compatibility between candidates at adjacent positions before lightweight greedy selection (Rusanovsky et al., 2026). These methods range from sequential correction heads to parallel or local scoring modules.

xPress also uses iterative parallel refinement: it augments an initial DFlash pass with a lightweight causal refiner and applies Jacobi-style logit corrections across the block (Wang et al., 2026a). Branch-Jacobi instead reruns the same prefix-trained drafter on independently packed moving-prefix rows and selects one complete linear proposal through first-MASK prefix-consistency checks. The two methods therefore incur different computation: xPress executes lightweight refiner iterations, whereas Branch-Jacobi executes additional draft forward passes. The information-floor analysis of Qiang et al. (2026) further shows that both missing path information and model quality under available conditions contribute to rejection.

CLLMs and Jacobi Forcing train language models on parallel-decoding trajectories to accelerate convergence (Kou et al., 2024; Hu et al., 2025). Branch-Jacobi instead refines an auxiliary drafter while keeping the target frozen. For parallel drafter training, D-PACE weights position-wise losses (Wu et al., 2026); VAT learns from simulated verification outcomes (Gu et al., 2026); Spec-AUF supervises the accepted prefix and the first failing position (Yang & Li, 2026). Our moving-anchor queries change the input condition by revealing a prefix for suffix prediction, and retain D-PACE suffix weighting.

2.3 Variable prefixes, draft trees, and verification budgets

DDTree constructs budgeted draft trees from DFlash marginals (Ringel & Romano, 2026); PRESTO adds prefix-aligned scoring (Wang et al., 2026b); JetSpec combines a causal parallel draft head with tree verification (Hu et al., 2026). D^2SD uses an initial diffusion drafter to identify likely rejection boundaries, a second variable-prefix diffusion drafter to fill alternative suffixes, and target tree attention to verify the resulting shared-prefix candidates (Zhang et al., 2026b). Branch-Jacobi trains one drafter for both initial and refined passes, evaluates live prefix-length branches on the draft side, and verifies only the selected sequence. It therefore shifts branching computation from target verification to draft refinement. DFlare increases drafter capacity through layer-wise target-feature fusion (Zhang et al., 2026a). DFlow reuses verifier representations from rejected suffixes (Zhang et al., 2026c), while ReTrace uses rejected draft states and target correction signals (Lin et al., 2026). Both transfer information across draft–verify cycles; our refinement occurs before target verification within a cycle.

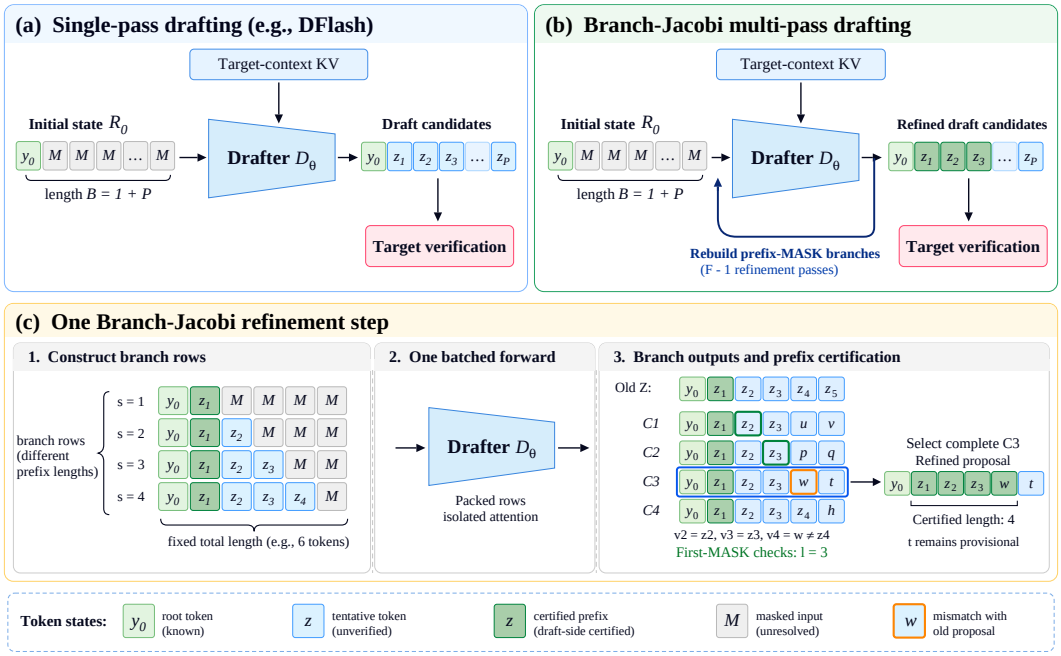

Figure 3: Branch-Jacobi drafting with packed prefix-MASK branches. First-MASK consistency checks identify a common prefix of the current proposal and the drafter-AR path. The selected branch supplies a suffix precomputed under the identified common prefix; green tokens indicate drafter-side certification, and the target verifier determines acceptance.

3 METHOD

Figure 3 summarizes the Branch-Jacobi drafting framework and one packed prefix-conditioned refinement step.

3.1 A PREFIX-CONDITIONAL BLOCK DRAFTER

Branch-Jacobi uses a frozen target model T and one trainable DFlash-style drafter D_θ. In each verification cycle, y_0 is the known root token following the verified context. A block has $B = P+1$ positions: the root and P proposal tokens. The fixed cycle condition $\mathcal{C}$ contains the verified context, target features available to the drafter, and logical positions and context state. For a prefix $u_{1:s}$, the drafter evaluates

$$R_s(u) = [y_0, u_1, \ldots, u_s, \mathrm{MASK}^{P-s}], \qquad 0 \le s < P. \tag{1}$$

Its masked positions produce proposal logits through the target embedding and output head inherited from DFlash (Chen et al., 2026). The block drafter retains bidirectional attention.

To state precisely what a prefix can certify, let $G_\theta(\mathcal{C}; R)_i$ be the vocabulary logits at proposal position i of row R. With a fixed total order $\prec$ for greedy ties, define the first-MASK decision

$$f_i(\mathcal{C}, u_{<i}) = \operatorname*{argmax}_{v \in \mathcal{V}}{}^{\prec} \left[G_\theta\big(\mathcal{C}; [y_0, u_{<i}, \mathrm{MASK}^{P-i+1}]\big)_i \right]_v, \quad i = 1, \ldots, P. \tag{2}$$

Positions after i contain a fixed MASK template, making equation 2 a function of the supplied prefix for fixed $\mathcal{C}$ and model parameters. We define the drafter-AR path $D = (d_1, \ldots, d_P)$ recursively as $d_i = f_i(\mathcal{C}, d_{<i})$, keeping target features fixed throughout the block. This triangular-recurrence/Jacobi viewpoint follows prior parallel-decoding work (Song et al., 2021; Santilli et al., 2023; Fu et al., 2024).

3.2 PACKED PREFIX-MASK BRANCHES

The first pass makes DFlash's root-only proposal $Z = (z_1, \ldots, z_P) = \text{Greedy } D_\theta(\mathcal{C}; R_0)$, where $R_0 = [y_0, \text{MASK}^P]$. The block is provisional, but $z_1 = d_1$ because its position-1 query defines f_1; hence the certified-prefix length starts at $c = 1$.

Given current proposal Z and certified prefix $Z_{1:c}$, a refinement pass uses equation 1 at every live anchor $s = c, \ldots, P-1$ to form

$$C_s = [Z_{1:s}, \text{ Greedy } D_\theta(\mathcal{C}; R_s(Z))_{s+1:P}], \qquad s = c, \ldots, P-1. \tag{3}$$

All $P-c$ rows are evaluated in one packed draft forward pass. Each row retains bidirectional within-block attention, and a block-diagonal mask isolates the rows. They share the same target context and logical block positions. Refinement stays within the drafter, and the target receives one linear proposal.

Packed-branch equivalence. Under the row-input, logical-position, target-context, attention, and normalization conditions of Lemma A.1 in Appendix A, packed and independent rows have identical exact-arithmetic logits, even with bidirectional within-row attention. Hence the first MASK of C_{i-1} equals $f_i(\mathcal{C}, Z_{<i})$.

3.3 COMMON-PREFIX CERTIFICATION AND BRANCH SELECTION

For $c < P$, let $v_i = C_{i-1,i} = f_i(\mathcal{C}, Z_{<i})$ be the first-MASK prediction for position $i = c+1, \ldots, P$, and define

$$V = (Z_{1:c}, v_{c+1}, \ldots, v_P), \qquad \ell = \text{LCP}(Z, V). \tag{4}$$

Here LCP denotes common-prefix length. Consecutive tests $v_i = z_i$ identify exactly the current proposal's common prefix with the drafter-AR path: $\ell = \text{LCP}(Z, D)$. We select

$$s^* = \min(\ell, P-1), \qquad Z^{\text{new}} = C_{s^*}, \qquad c^{\text{new}} = \min(\ell+1, P). \tag{5}$$

If the first mismatch occurs at $j = \ell + 1$, branch C_ℓ retains the matching prefix and predicts d_j at its first MASK, certifying one additional token. If all tests pass, $\ell = P$ and $C_{P-1} = Z$ certifies the entire proposal. When $c = P$, refinement preserves (Z, c).

All rows use the same pre-pass Z and are evaluated in parallel. Each row supplies a first-MASK consistency check and predictions for its entire masked suffix. At the first mismatch $\ell + 1$, selecting C_ℓ retrieves the suffix already computed under the matching prefix $Z_{1:\ell}$, without an additional post-selection draft call. Entries of V beyond $\ell + 1$ use old prefixes containing the mismatching token $z_{\ell+1}$. In this mismatch case, the selector returns the complete branch C_ℓ, whose suffix was computed under $Z_{1:\ell}$. Its suffix remains provisional: the newly certified token and later tokens are predicted in the same forward pass, so the later predictions do not condition on that new token. Algorithm 1 specifies the fixed pass budget; Appendix A gives examples and the common-prefix characterization.

Algorithm 1 Common-prefix certification in one target cycle

Require: Fixed $\mathcal{C}$, root y_0, D_θ, T, width P, pass budget F
$Z \leftarrow \text{Greedy } D_\theta(\mathcal{C}; [y_0, \text{MASK}^P])$; $c \leftarrow 1$
for $r = 2, \ldots, F$ **do**
  Obtain $\{C_s\}$ by evaluating rows $R_s(Z)$, $s = c, \ldots, P-1$, with fixed-shape padding
  if $c < P$ **then**
    $\ell \leftarrow c$
    for $i = c+1, \ldots, P$ **do**
      if $C_{i-1,i} \neq z_i$ **then**
        break
      end if
      $\ell \leftarrow i$
    end for
    $s^* \leftarrow \min(\ell, P-1)$
    $Z \leftarrow C_{s^*}$; $c \leftarrow \min(\ell+1, P)$
  end if
end for
$(a, b) \leftarrow \text{LinearVerify}_T(y_0, Z)$
return $Z_{1:a}$ and target bonus token b

Proposition 1 (common-prefix identification and sound extension). Assume deterministic greedy drafter decisions, fixed $\mathcal{C}$ and model parameters, and packed/independent branch equivalence. If $Z_{1:c} = D_{1:c}$ and $c < P$, then equation 4 yields $\ell = \mathrm{LCP}(Z, D)$, and the selected branch satisfies $Z^{\mathrm{new}}_{1:c^{\mathrm{new}}} = D_{1:c^{\mathrm{new}}}$ with $c^{\mathrm{new}} = \min(\ell + 1, P)$.

Proof. Starting from $Z_{1:c} = D_{1:c}$, each passed test gives $z_i = v_i = f_i(\mathcal{C}, Z_{<i}) = f_i(\mathcal{C}, D_{<i}) = d_i$. At the first failed test $j = \ell+1$, the preceding prefixes still match, so $v_j = d_j \neq z_j$ and ℓ is the exact common-prefix length. Branch C_ℓ copies $Z_{1:\ell} = D_{1:\ell}$ and supplies $v_j = d_j$, proving the extension. If all tests pass, $Z = D$ and the last branch C_{P-1} also equals D. Initialization certifies $c = 1$ by construction. □

The LCP identity applies to the pre-pass proposal Z. Certification covers the first c^{new} positions of the selected proposal; later positions may also match D. Target verification determines acceptance. Drafter decisions remain greedy at both target temperatures, so the argument also applies when the target samples.

3.4 Target verification and computation

The target verifies the selected linear sequence $[y_0, Z_{1:P}]$. In greedy decoding, it commits the longest target-matching prefix $Z_{1:a}$ and a target bonus token, advancing $a + 1$ positions. At temperature 1, lossless speculative sampling uses delta proposals $q_i = \delta_{Z_i}$, standard draft–target acceptance probabilities, and exact residual sampling upon rejection, preserving the target's full-softmax distribution (Leviathan et al., 2023; Chen et al., 2023).

Branch-Jacobi with pass budget F executes one initial pass and $F - 1$ packed refinement passes with the same drafter; we evaluate $F \in \{1, \ldots, 5\}$. The measured implementation keeps fixed physical row shapes. Once $c = P$, subsequent passes preserve the proposal while still executing the scheduled forwards.

3.5 Moving-anchor training and causal prefix supervision

Moving-anchor training exposes the drafter to the prefix-MASK queries used during refinement. For each ground-truth block $y_{0:P}$, we evaluate $R_k(y)$ from equation 1 with $k \in \{0, \ldots, P - 1\}$. We define $m_i = \mathbf{1}\{i > k \text{ and } i \text{ is a valid response target}\}$ and apply the main suffix loss only where $m_i = 1$. We assign k cyclically across valid sampled blocks, approximately balancing prefix lengths; the endpoints retain the root-only task ($k = 0$) and one prediction slot ($k = P - 1$).

Suffix training retains block-isolated bidirectional attention. A masked query sees the revealed prefix and pre-root target context, never its label, unrevealed suffix tokens, the original target sequence at or after the root, or other blocks. Inference uses the same row form with proposed prefixes; the auxiliary objective below trains prefix generation.

For the masked suffix, we follow D-PACE's continuation-weighted cross-entropy idea (Wu et al., 2026) with smoothing $\alpha = 0.5$, applied only where $m_i = 1$. Appendix A.1 gives the objective and its normalization.

For each revealed token y_i, $1 \leq i \leq k$, our causal prefix objective independently queries $[y_0, y_{1:i-1}, \mathrm{MASK}_i] \to y_i$. With N denoting the number of sequences in the loss normalization, the auxiliary and total losses are

$$\mathcal{L}_{\mathrm{prefix}} = \frac{1}{N} \sum_{\text{blocks}} \sum_{i=1}^{k} \mathrm{CE}(D_\theta([y_0, y_{1:i-1}, \mathrm{MASK}_i]), y_i)\,, \qquad \mathcal{L} = \mathcal{L}_{\mathrm{suffix}} + \lambda \mathcal{L}_{\mathrm{prefix}}. \tag{6}$$

The auxiliary queries share prefix states under block-isolated causal attention. Query i sees preceding content and its own MASK; suffix training and packed inference retain bidirectional attention. These queries add no inference parameters. Appendix A.1 gives the attention details; we use $\lambda = 0.2$ for the main configuration and examine its sensitivity in Appendix B.

Table 1: Decode-stage speedup over AR and average acceptance length (τ, including the target bonus token). Parentheses give EAGLE-3 tree size or DFlash/B-Jacobi block size; F counts draft passes. Avg. reports geometric-mean speedup and arithmetic-mean τ over seven benchmarks. Timing is defined in Appendix D.

Model	Method	MATH						CODE						CHAT		*Avg.*	
		GSM8K		MATH-500		AIME25		HumanEval		MBPP		LCB		MT-Bench		*Avg.*	
	Temperature = 0	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ
Q3-4B	EAGLE-3 (16)	2.03×	3.29	1.90×	3.09	1.91×	3.09	1.89×	3.06	1.81×	2.94	1.76×	2.88	1.82×	3.02	1.87×	3.05
	EAGLE-3 (60)	2.32×	3.77	2.17×	3.54	2.19×	3.55	2.12×	3.46	2.07×	3.38	1.96×	3.25	2.09×	3.47	2.13×	3.49
	DFlash (16)	5.24×	6.51	6.14×	7.79	5.76×	7.28	5.23×	6.61	4.84×	6.13	5.39×	6.91	2.87×	4.35	4.95×	6.51
	B-Jacobi (16, $F=1$)	5.56×	6.69	6.25×	7.65	5.77×	6.96	5.10×	6.11	4.78×	5.77	5.74×	7.13	3.09×	4.43	5.08×	6.39
	B-Jacobi (16, $F=2$)	6.66×	8.29	7.35×	9.27	7.68×	8.50	5.99×	7.49	5.74×	7.11	7.15×	8.59	3.55×	5.34	6.14×	7.80
	B-Jacobi (16, $F=3$)	7.04×	8.80	7.59×	9.79	8.11×	9.01	6.18×	7.97	5.95×	7.61	7.16×	9.02	3.63×	5.66	**6.35**×	8.27
	B-Jacobi (16, $F=4$)	6.11×	8.95	7.65×	9.94	7.86×	9.17	5.94×	8.13	5.94×	7.80	6.71×	9.12	3.64×	5.77	6.11×	8.41
	B-Jacobi (16, $F=5$)	6.63×	8.98	7.20×	9.97	7.92×	9.23	6.08×	8.17	5.76×	7.86	7.04×	9.14	3.48×	5.80	6.13×	**8.45**
Q3-8B	EAGLE-3 (16)	1.98×	3.22	1.87×	3.05	1.87×	3.01	1.94×	3.16	1.74×	2.82	1.61×	2.65	1.70×	2.84	1.81×	2.97
	EAGLE-3 (60)	2.30×	3.73	2.15×	3.50	2.19×	3.58	2.24×	3.65	2.00×	3.24	1.82×	3.04	1.94×	3.26	2.09×	3.43
	DFlash (16)	5.26×	6.56	6.23×	7.93	5.85×	7.42	5.25×	6.53	4.72×	5.96	5.51×	7.20	2.79×	4.23	4.95×	6.55
	B-Jacobi (16, $F=1$)	5.70×	6.84	6.55×	8.01	6.14×	7.46	5.26×	6.33	4.85×	5.86	5.97×	7.54	3.06×	4.39	5.23×	6.63
	B-Jacobi (16, $F=2$)	6.88×	8.43	7.65×	9.56	7.52×	8.97	6.00×	7.73	5.57×	7.25	6.61×	8.97	3.79×	5.24	6.15×	8.02
	B-Jacobi (16, $F=3$)	7.15×	9.00	7.92×	10.05	7.62×	9.47	6.25×	8.27	5.96×	7.81	7.10×	9.42	3.86×	5.51	**6.40**×	8.50
	B-Jacobi (16, $F=4$)	7.02×	9.16	8.01×	10.20	7.51×	9.60	6.05×	8.45	5.96×	7.92	7.00×	9.58	3.84×	5.59	6.33×	8.64
	B-Jacobi (16, $F=5$)	6.94×	9.20	7.71×	10.25	7.46×	9.64	6.06×	8.50	5.81×	7.96	6.84×	9.62	3.74×	5.61	6.22×	**8.68**
	Temperature = 1	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ	Speedup	τ
Q3-4B	EAGLE-3 (16)	1.97×	3.22	1.82×	2.98	1.70×	2.79	1.83×	3.00	1.77×	2.91	1.70×	2.80	1.74×	2.93	1.79×	2.95
	EAGLE-3 (60)	2.20×	3.68	2.05×	3.42	1.89×	3.17	2.03×	3.41	1.97×	3.32	1.86×	3.16	1.94×	3.35	1.99×	3.36
	DFlash (16)	4.45×	5.68	4.90×	6.47	3.90×	5.17	4.47×	5.69	4.01×	5.14	4.67×	6.21	2.46×	3.77	4.04×	5.45
	B-Jacobi (16, $F=1$)	5.09×	6.18	5.22×	6.55	3.86×	4.88	4.62×	5.57	4.42×	5.35	5.41×	6.80	2.90×	4.13	4.42×	5.64
	B-Jacobi (16, $F=2$)	5.96×	7.51	5.81×	7.55	4.08×	5.33	5.39×	6.76	5.10×	6.41	6.23×	8.12	3.18×	4.81	4.99×	6.64
	B-Jacobi (16, $F=3$)	6.18×	8.01	5.91×	7.85	3.93×	5.38	5.57×	7.12	5.28×	6.84	6.30×	8.48	3.20×	5.09	**5.06**×	6.97
	B-Jacobi (16, $F=4$)	6.07×	8.06	5.66×	7.81	4.12×	5.72	5.51×	7.26	5.26×	7.01	6.24×	8.58	3.10×	5.04	5.01×	7.07
	B-Jacobi (16, $F=5$)	5.91×	8.04	5.53×	7.81	4.03×	5.79	5.36×	7.28	5.13×	7.01	6.10×	8.57	3.08×	5.08	4.90×	**7.08**
Q3-8B	EAGLE-3 (16)	1.89×	3.12	1.75×	2.88	1.65×	2.71	1.86×	3.04	1.69×	2.75	1.55×	2.55	1.58×	2.66	1.71×	2.82
	EAGLE-3 (60)	2.15×	3.61	1.99×	3.33	1.83×	3.08	2.08×	3.50	1.91×	3.19	1.70×	2.88	1.77×	3.06	1.91×	3.24
	DFlash (16)	4.38×	5.56	4.74×	6.21	3.75×	4.84	4.23×	5.32	3.84×	4.86	4.58×	6.06	2.39×	3.58	3.90×	5.20
	B-Jacobi (16, $F=1$)	5.15×	6.23	5.09×	6.43	3.86×	4.83	4.46×	5.35	4.23×	5.13	5.57×	7.10	2.75×	3.93	4.35×	5.57
	B-Jacobi (16, $F=2$)	5.80×	7.32	5.76×	7.45	4.33×	5.60	4.97×	6.24	4.80×	6.03	6.36×	8.41	2.91×	4.44	4.86×	6.50
	B-Jacobi (16, $F=3$)	5.95×	7.73	5.74×	7.62	4.28×	5.78	5.03×	6.47	5.00×	6.51	6.45×	8.80	2.99×	4.78	**4.93**×	6.81
	B-Jacobi (16, $F=4$)	5.89×	7.83	5.71×	7.80	4.23×	5.75	5.03×	6.60	4.96×	6.58	6.30×	8.83	2.94×	4.86	4.88×	6.89
	B-Jacobi (16, $F=5$)	5.79×	7.85	5.52×	7.74	4.19×	5.88	4.95×	6.66	4.90×	6.64	6.20×	8.86	2.87×	4.81	4.79×	**6.92**

4 EXPERIMENTS

4.1 EVALUATION SETUP

We evaluate Qwen3-4B and Qwen3-8B (Yang et al., 2025) with thinking mode disabled. The benchmark suite covers mathematical reasoning (GSM8K, MATH-500, and AIME25) (Cobbe et al., 2021; Lightman et al., 2024; Mathematical Association of America, 2025), code generation (HumanEval, MBPP, and LiveCodeBench) (Chen et al., 2021; Austin et al., 2021; Jain et al., 2024), and conversational generation (MT-Bench) (Zheng et al., 2023). We compare Branch-Jacobi with EAGLE-3 and DFlash under greedy decoding ($T = 0$) and temperature-1 sampling. All evaluations measure single-request decode latency on one H100 with batch size and concurrency one; DFlash and Branch-Jacobi use block size 16.

We train DFlash-style drafters with moving-anchor suffix queries, D-PACE continuation weighting, and causal prefix supervision. Appendix D specifies the training data, model configurations, and inference and timing protocols; Appendix C describes the matched training and inference controls.

For each response, time per output token is $\mathrm{TPOT} = t_{\mathrm{dec}}/N$, where t_{dec} is measured decode time and N counts all returned tokens, including the first; Appendix D defines the timing boundaries. Dataset speedup is the ratio of mean target-only to mean speculative TPOT. Acceptance length includes the target bonus token and is averaged over responses. Across datasets, we report geometric-mean speedup and arithmetic-mean acceptance length. The pass budget F counts all draft forwards per cycle, including the initial proposal. A fixed-checkpoint F sweep isolates inference refinement without retraining.

4.2 Main results

Table 1 compares autoregressive and block-parallel drafting across both target sizes and decoding regimes. Increasing EAGLE-3's tree size from 16 to 60 improves speedup and acceptance length; DFlash achieves higher speedup through one-pass block drafting. With $F \geq 2$, Branch-Jacobi improves speedup and acceptance length over DFlash on every benchmark and in all four model–temperature aggregates. The pass-budget sweep holds each Branch-Jacobi checkpoint fixed, isolating the inference-time refinement procedure from changes in training. The table reports the change in geometric-mean speedup as the pass budget increases. Matched five-layer training controls isolate query geometry (Table 5), while a shared three-layer checkpoint controls the branch-count comparison (Table 6). These within-configuration comparisons complement the complete-system comparison with DFlash.

For both Qwen3-4B and Qwen3-8B, three passes achieve the highest measured aggregate speedup among the tested budgets at $T = 0$ and $T = 1$. Five passes yield the highest aggregate AAL. The acceptance-maximizing and latency-minimizing budgets therefore differ in the completed main evaluation.

4.3 Runtime and prefix behavior

Figure 1 and Tables 7–8 examine whether acceptance gains compensate for additional draft cost. On Qwen3-4B/MATH-500, increasing F from 1 to 4 raises AAL from 7.650 to 9.936 and reduces TPOT from 4.312 to 3.529 ms. A fifth pass raises AAL only to 9.972 while increasing TPOT to 3.750 ms. The $F = 4$ versus $F = 3$ TPOT difference is small and its paired-response 95% interval includes zero (Appendix D).

The right panel of Figure 4 tests whether drafter-side certification is informative about target acceptance. Table 9 shows positive rank correlations, decreasing from 0.702 at $F = 2$ to 0.417 at $F = 5$. Over these budgets, the cycle-weighted mean certified length rises from 8.281 to 14.124, while mean target-accepted draft tokens rise from 8.033 to 8.722. Certification tracks the drafter path and is positively associated with target acceptance; target verification determines the committed tokens.

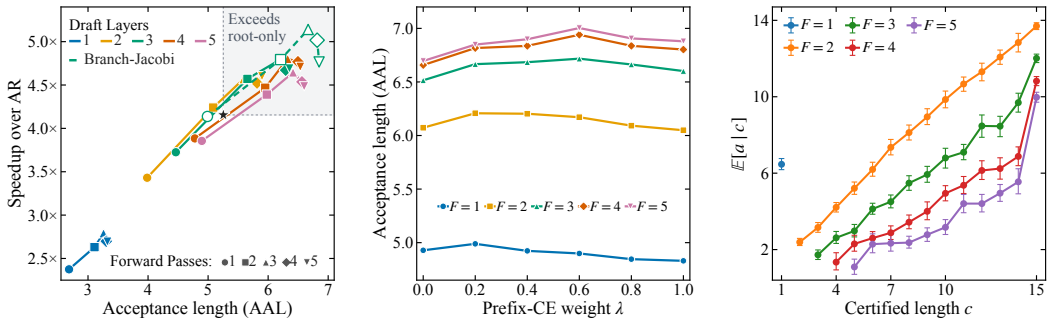

Figure 4: Qwen3-4B ablations and prefix diagnostics at $T = 0$. Left and center use 100K-subset training and 128 GSM8K prompts. Left: DFlash-loss depth sweep and Branch-Jacobi; the star is the five-layer root-only control from Table 5. Center: prefix-loss weight. Right: target acceptance by certified prefix length for the full-data checkpoint on 128 MATH-500 prompts, with 95% response-bootstrap intervals.

4.4 Ablation studies

All four ablations use 128 GSM8K prompts, Qwen3-4B at $T = 0$, and the 100K training subset. Appendix B extends the depth and prefix-loss studies to HumanEval with the same checkpoints and inference settings; the training-query and branch-count controls remain GSM8K-only. Figure 4 compares the DFlash-loss depth sweep, the five-layer root-only control, and three-layer Branch-Jacobi (left), and varies λ with drafter and continuation weighting fixed (center). The additional Branch-Jacobi series evaluates a joint change in training objective and refinement implementation relative to the depth sweep (Appendix D).

Draft depth and pass budget. The depth study contrasts additional model depth with additional refinement passes as ways to allocate draft computation. In Table 3, refinement raises AAL at every tested depth, with depth-dependent gains in speedup. On GSM8K, at $F = 3$, the three-layer drafter achieves $4.78\times$ speedup and 6.16 AAL, compared with $4.65\times$ and 6.41 for five layers. Thus, a deeper drafter can accept more tokens while costing more time per returned token.

Prefix-loss weight. The prefix-loss study varies $\lambda \in \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ while holding the other training settings fixed. At $F = 4$, all nonzero weights yield higher GSM8K AAL than $\lambda = 0$: 6.80–6.94 versus 6.66 (Table 4). The main configuration, $\lambda = 0.2$, reaches 6.82; $\lambda = 0.6$ has the highest AAL in this sweep. These results support causal prefix supervision for multi-pass acceptance.

Controlled training and inference. Two controls test the training and inference design on GSM8K. Matched five-layer drafters compare root-only and moving-anchor training; a shared three-layer drafter compares one active branch with the packed family. Table 2 summarizes $F = 1$ and $F = 4$, and Appendix C provides all pass budgets. The moving-anchor and packed-family entries reuse the corresponding depth-study measurements. These controls quantify training-query effects with matched five-layer drafters and branch-count effects with a shared three-layer checkpoint.

Table 2: Controlled comparisons on GSM8K with Qwen3-4B at $T = 0$ using the 100K training subset. Speedup is the ratio of mean TPOTs; AAL is averaged over responses. Training-query controls use matched five-layer drafters; branch-count controls share a three-layer checkpoint. Compare configurations within each group.

	$F = 1$		$F = 4$	
Configuration	Speedup	AAL	Speedup	AAL
Training query geometry (5 layers)				
Root-only	$4.15\times$	5.25	$2.94\times$	4.02
Moving-anchor	$3.86\times$	4.90	$4.54\times$	6.56
Inference branches (shared 3-layer checkpoint)				
Single branch ($K = 1$)	$3.71\times$	4.46	$4.15\times$	5.44
Packed branches ($K = 14$)	$3.72\times$	4.46	$4.67\times$	6.29

Root-only training has the stronger one-pass result but degrades under refinement, whereas moving-anchor training improves both AAL and speedup. Packed branches likewise outperform one active branch at $F = 4$. These GSM8K controls support training-query alignment and access to multiple anchors (Table 2).

5 Conclusion

We introduced Branch-Jacobi, which restores within-block prefix conditioning to a DFlash-style drafter through packed multi-pass refinement. First-MASK consistency checks identify a common prefix with the drafter-autoregressive path, and selecting one complete branch extends certification before linear target verification. Moving-anchor training and causal prefix supervision align the drafter with these queries. Across seven benchmarks on Qwen3-4B and Qwen3-8B, Branch-Jacobi improves acceptance length and decode-stage speedup over DFlash under greedy and temperature-1 decoding. The GSM8K controls support moving-anchor training and packed branches for refinement. Across the four main model–temperature settings, three passes give the highest measured aggregate speedup, while five passes give the highest aggregate acceptance length. The pass budget therefore balances acceptance gains against additional draft computation.

Our evaluation characterizes single-request decode-stage latency. Future work can extend this evaluation to full-request latency and concurrent serving, and study pass-budget selection across hardware, block sizes, and model architectures. Drafter-side certification constrains the proposal path, and target verification preserves the target distribution.

AI USE STATEMENT

We used large language models to assist with writing and polishing the manuscript. We also used them for retrieval and discovery of relevant literature, including related work.

ETHICS STATEMENT

This work studies the computational efficiency of language-model decoding. Generated outputs retain the underlying models' risks of harmful, biased, or privacy-sensitive content. The target-distribution analysis addresses decoding correctness. Use of the referenced models and datasets remains subject to their applicable terms and restrictions.

REPRODUCIBILITY STATEMENT

Section 3 specifies the drafting, certification, verification, and training procedures. Appendix A provides the assumptions and proofs, together with the detailed loss formulation. Appendices B and C report the GSM8K and HumanEval depth and prefix-loss ablations, together with the GSM8K training and inference controls. Appendix D documents the training configuration, public baseline checkpoints, evaluation subsets, inference settings, timing boundaries, normalization, and uncertainty estimates.

REFERENCES

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.

Sahil Chaudhary. Code alpaca: An instruction-following LLaMA model for code generation. https://github.com/sahil280114/codealpaca, 2023.

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, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. URL https://arxiv.org/abs/2107.03374.

Xin Cheng, Xingkai Yu, Chenze Shao, Jiashi Li, Yunfan Xiong, Yi Qian, Jiaqi Zhu, Shirong Ma, Xiaokang Zhang, Jiasheng Ye, Qinyu Chen, Chengqi Deng, Jiping Yu, Damai Dai, Zhengyan Zhang, Yixuan Wei, Yixuan Tan, Wenkai Yang, Runxin Xu, Yu Wu, Zhean Xu, Xuanyu Wang, Muyang Chen, Rui Tian, Xiao Bi, Zhewen Hao, Shaoyuan Chen, Huanqi Cao, Wentao Zhang, Anyi Xu, Huishuai Zhang, Dongyan Zhao, and Wenfeng Liang. DSpark: Confidence-scheduled speculative decoding with semi-autoregressive generation. *arXiv preprint arXiv:2607.05147*, 2026. URL https://arxiv.org/abs/2607.05147.

Jacob K. Christopher, Brian R. Bartoldson, Tal Ben-Nun, Michael Cardei, Bhavya Kailkhura, and Ferdinando Fioretto. Speculative diffusion decoding: Accelerating language generation through diffusion. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 12042–12059, 2025. doi: 10.18653/v1/2025.naacl-long.601. URL https://aclanthology.org/2025.naacl-long.601/.

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.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. Break the sequential dependency of LLM inference using lookahead decoding. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 14060–14079, 2024. URL https://proceedings.mlr.press/v235/fu24a.html.

Geonmo Gu, Byeongho Heo, HeeJae Jun, Yoohoon Kang, Sangmin Lee, Sangdoo Yun, and Dongyoon Han. Verification-Aware Training for Speculative Decoding. *arXiv preprint arXiv:2608.30135*, 2026. URL https://arxiv.org/abs/2608.30135v1. Version v1.

Lanxiang Hu, Siqi Kou, Yichao Fu, Samyam Rajbhandari, Tajana Rosing, Yuxiong He, Zhijie Deng, and Hao Zhang. Fast and Accurate Causal Parallel Decoding using Jacobi Forcing. *arXiv preprint arXiv:2512.14681*, 2025. URL https://arxiv.org/abs/2512.14681v1. Version v1.

Lanxiang Hu, Zhaoxiang Feng, Yulun Wu, Haoran Yuan, Yujie Zhao, Yu-Yang Qian, Bojun Wang, Peng Zhao, Daxin Jiang, Yibo Zhu, Tajana Rosing, and Hao Zhang. JetSpec: Breaking the Scaling Ceiling of Speculative Decoding with Parallel Tree Drafting. *arXiv preprint arXiv:2606.18394*, 2026. URL https://arxiv.org/abs/2606.18394v3. Version v3.

Ailin Huang, Ang Li, Aobo Kong, Bin Wang, Binxing Jiao, Bo Dong, Bojun Wang, Boyu Chen, Brian Li, Buyun Ma, et al. Step 3.5 flash: Open frontier-level intelligence with 11b active parameters. *arXiv preprint arXiv:2602.10604*, 2026a. URL https://arxiv.org/abs/2602.10604.

Jianuo Huang, Yaojie Zhang, Qituan Zhang, Hao Lin, Hanlin Xu, and Linfeng Zhang. Domino: Decoupling causal modeling from autoregressive drafting in speculative decoding. *arXiv preprint arXiv:2605.29707*, 2026b. URL https://arxiv.org/abs/2605.29707.

Inco AI. DFlash 2: Keep drafting parallel, August 2026. URL https://inco.ai/blog/dflash2/. Official technical blog, August 18, 2026.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024. URL https://arxiv.org/abs/2403.07974.

Siqi Kou, Lanxiang Hu, Zhezhi He, Zhijie Deng, and Hao Zhang. CLLMs: Consistency large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 25426–25440, 2024. URL https://proceedings.mlr.press/v235/kou24a.html.

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.

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, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/c7b5a35ea98b62512a869c19ea7b03cb-Abstract-Conference.html.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Luxi Lin, Zhanpeng Zeng, Shuang Peng, Songwei Liu, and Rongrong Ji. ReTrace: Rejected-Trajectory Conditioning for Speculative Decoding. *arXiv preprint arXiv:2608.29748*, 2026. URL https://arxiv.org/abs/2608.29748v2. Version v2.

Mathematical Association of America. American invitational mathematics examination 2025. American Mathematics Competitions, 2025.

Dhruv Nathawani, Shuoyang Ding, Vitaly Lavrukhin, Igor Gitman, Somshubra Majumdar, Evelina Bakhturina, Boris Ginsburg, and Jane Polak Scowcroft. Nemotron-Post-Training-Dataset-v2, 2025. URL https://huggingface.co/datasets/nvidia/Nemotron-Post-Training-Dataset-v2.

Xinwei Qiang, Xiang Fang, Chang Chen, Zaifeng Pan, Yue Guan, and Yufei Ding. Beyond parallel blindness: Information floors and model gaps in block drafting. *arXiv preprint arXiv:2608.27339*, 2026. URL https://arxiv.org/abs/2608.27339v3. Version v3.

Peer Rheinboldt, Frédéric Berdoz, and Roger Wattenhofer. TreeFlash: Parallel AR-approximation for faster speculative decoding. *arXiv preprint arXiv:2606.03819*, 2026. URL https://arxiv.org/abs/2606.03819.

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.

Matan Rusanovsky, Yoav Miron, Roy Uziel, Omer Belhasin, Hao Guo, Ran Zilberstein, Maor Ashkenazi, and Michael Elad. LiLiCorr: Lightweight Likelihood Correlation of Parallel Drafts for Speculative Decoding. *arXiv preprint arXiv:2608.20530*, 2026. URL https://arxiv.org/abs/2608.20530v2. Version v2.

Andrea Santilli, Silvio Severino, Emilian Postolache, Valentino Maiorca, Michele Mancusi, Riccardo Marin, and Emanuele Rodola. Accelerating transformer inference for translation via parallel decoding. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12336–12355, 2023. doi: 10.18653/v1/2023.acl-long.689. URL https://aclanthology.org/2023.acl-long.689/.

Yang Song, Chenlin Meng, Renjie Liao, and Stefano Ermon. Accelerating feedforward computation via parallel nonlinear equation solving. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9791–9800, 2021. URL https://proceedings.mlr.press/v139/song21a.html.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. Blockwise parallel decoding for deep autoregressive models. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/c4127b9194fe8562c64dc0f5bf2c93bc-Abstract.html.

Zheng Wang, Davis Wertheimer, Yu Chin Fabian Lim, Mudhakar Srivatsa, Raghu K. Ganti, Minjia Zhang, and Naigang Wang. xPress: Parallel refinement for diffusion drafters in speculative decoding. *arXiv preprint arXiv:2608.02438*, 2026a. URL https://arxiv.org/abs/2608.02438.

Zheng Wang, Zhifan Ye, Qi Cheng, Yonggan Fu, Ziyan Wang, Feng Zhu, Haozhe Zhao, Jan Kautz, Pavlo Molchanov, Humphrey Shi, and Minjia Zhang. PRESTO: Prefix-Aligned Tree Drafting for Diffusion Speculative Decoding. *arXiv preprint arXiv:2607.22634*, 2026b. URL https://arxiv.org/abs/2607.22634v1. Version v1.

Tianyu Wu, Yu Yao, Zhenting Qi, Han Zheng, Zhuohan Wang, Haoran Ma, Lawrence Liao, Himabindu Lakkaraju, Ju Li, and Yilun Du. D-PACE: Dynamic position-aware cross-entropy for parallel speculative drafting. *arXiv preprint arXiv:2605.18810*, 2026. URL https://arxiv.org/abs/2605.18810.

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

Tianjian Yang and Meng Li. Spec-AUF: Accept-Until-Fail Training under Train-Inference Misalignment for Masked Block Drafters. *arXiv preprint arXiv:2607.01893*, 2026. URL https://arxiv.org/abs/2607.01893v1. Version v1.

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.

Liyuan Zhang, Jiarui Zhang, Jinwei Yao, Ran Yan, Yuchen Yang, Jiahao Zhang, Tongkai Yang, Yi Wu, and Binhang Yuan. D^{2}SD: Accelerating speculative decoding with dual diffusion draft models. *arXiv preprint arXiv:2606.04446*, 2026b. URL https://arxiv.org/abs/2606.04446.

Yaojie Zhang, Linfeng Zhang, Bin Cui, and Xupeng Miao. DFlow: Enabling verifier information flow in block diffusion speculative decoding. *arXiv preprint arXiv:2609.06498*, 2026c. URL https://arxiv.org/abs/2609.06498.

Hong-Kai Zheng and Piji Li. DeLS-Spec: Decoupled long-short contexts for parallel speculative drafting. *arXiv preprint arXiv:2607.07409*, 2026. URL https://arxiv.org/abs/2607.07409.

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, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

A METHOD DETAILS AND PROOFS

Concrete packed-prefix and selection examples. For $P = 4$, the packed prefix-MASK branches can be written directly as

$$\begin{aligned}
\text{initial:}\quad & [y_0, \mathrm{M}, \mathrm{M}, \mathrm{M}, \mathrm{M}] && \longrightarrow Z = (z_1, z_2, z_3, z_4),\\
s = 1:\quad & [y_0, z_1, \mathrm{M}, \mathrm{M}, \mathrm{M}] && \longrightarrow C_1 = (z_1, \boldsymbol{v_2}, *, *),\\
s = 2:\quad & [y_0, z_1, z_2, \mathrm{M}, \mathrm{M}] && \longrightarrow C_2 = (z_1, z_2, \boldsymbol{v_3}, *),\\
s = 3:\quad & [y_0, z_1, z_2, z_3, \mathrm{M}] && \longrightarrow C_3 = (z_1, z_2, z_3, \boldsymbol{v_4}).
\end{aligned}$$

Here $\mathrm{M} = \mathrm{MASK}$, bold tokens are first-MASK decisions, and $*$ marks provisional suffix predictions. If $c = 1$, $v_2 = z_2$, and $v_3 \neq z_3$, then $\ell = 2$: the selector returns the complete C_2 and certifies positions 1–3, including its new token v_3. No agreement with the provisional position-3 output of C_1 is required. The suffix marked $*$ remains provisional.

Lemma A.1 (independent-row equivalence of packed branches). Suppose that a packed branch and its independently evaluated counterpart have the same token/MASK inputs, logical positions, fixed target features and context states, model parameters, attention-visible key/value nodes with the same multiplicities and biases, and no cross-branch attention or normalization. In exact arithmetic, their logits at every corresponding node are equal.

Proof. At the input layer, corresponding token embeddings, position encodings, and condition features coincide. Assume corresponding hidden states agree at layer ℓ. Each corresponding query sees the same key/value multiset and attention bias, so its attention scores, per-query normalization, and attention output coincide. Per-node normalization, residual connections, and feed-forward maps produce the same layer-$\ell + 1$ hidden state. Induction over layers gives identical final hidden states and logits. Identical tie-breaking gives identical greedy first-MASK outputs. The argument allows within-row bidirectional attention: later fixed MASK nodes are present in both computations. □

Proposition A.2 (common-prefix characterization for a fixed candidate). Fix a candidate $X = (x_1, \ldots, x_P)$ *before* the parallel evaluation. Evaluate the independent prefix row $[y_0, x_{<i}, \mathrm{MASK}^{P-i+1}]$ for each position i, and let $\hat{v}_i$ be its first-MASK greedy output. Under Lemma A.1,

$$\hat{v}_i = f_i(\mathcal{C}, x_{<i}). \tag{7}$$

If $X_{1:c} = D_{1:c}$, then for every $m \in \{0, \ldots, P - c\}$,

$$\left[\, \forall i = c+1, \ldots, c+m,\ x_i = \hat{v}_i \,\right] \quad \Longleftrightarrow \quad X_{c+1:c+m} = D_{c+1:c+m}. \tag{8}$$

Consequently, if r is the longest consecutive number of passed tests from $c+1$, then

$$r = \mathrm{LCP}(X, D) - c. \tag{9}$$

Proof. For $m = 0$, both sides of equation 8 hold vacuously. For $m \geq 1$, suppose first that each position $c+1, \ldots, c+m$ passes. The known prefix gives $x_{<c+1} = d_{<c+1}$ and therefore

$$x_{c+1} = \hat{v}_{c+1} = f_{c+1}(\mathcal{C}, d_{<c+1}) = d_{c+1}. \tag{10}$$

If $x_{<i} = d_{<i}$ at a later position i, its passed test implies

$$x_i = \hat{v}_i = f_i(\mathcal{C}, x_{<i}) = f_i(\mathcal{C}, d_{<i}) = d_i. \tag{11}$$

Induction gives the right side of equation 8. Conversely, if the right side holds, then $X_{1:c} = D_{1:c}$ makes $x_{<i} = d_{<i}$ for every evaluated i, and

$$\hat{v}_i = f_i(\mathcal{C}, x_{<i}) = f_i(\mathcal{C}, d_{<i}) = d_i = x_i. \tag{12}$$

Thus every position passes. The set of passed lengths is exactly the set of common-prefix extension lengths, proving equation 9. All prefix rows were fixed from X before the forward; they can therefore be evaluated in parallel without depending on one another's new outputs. □

Selector output and fixed-candidate common prefixes. Applying Proposition A.2 with $X = Z$ identifies the pre-pass proposal's exact common-prefix length. Selecting a branch changes the proposal, whose certified prefix is characterized by Proposition 1. Take $P = 3$, old $Z = (A, A, A)$, $c = 1$, and branch outputs $C_1 = (A, B, C)$, $C_2 = (A, A, E)$. The first test gives $v_2 = B \neq z_2 = A$, so $\ell = 1$, $S = C_1$, and $c' = 2$. Yet $f_1(\mathcal{C}, \emptyset) = A$, $f_2(\mathcal{C}, A) = B$, and $f_3(\mathcal{C}, A, B) = C$ can give $D = S$, with $\mathrm{LCP}(S, D) = 3 > c'$. The selector queried the old-Z prefixes; evaluating S at position 3 would use the prefix (A, B). Proposition 1 guarantees $c' \leq \mathrm{LCP}(S, D)$; this example gives strict inequality.

A.1 Continuation-weighted suffix loss and auxiliary attention

Following D-PACE (Wu et al., 2026), we weight valid masked-suffix positions using a surrogate for consecutive-prefix correctness. For the training row $R_k(y)$ and supervision mask m_i in Section 3.5, let $\ell_i = -\log p_\theta(y_i \mid \mathcal{C}, R_k)$ and $p_i = \exp(-\ell_i)$. With $\alpha = 0.5$, the exact objective is

$$\bar{s}_i = \begin{cases} \alpha + (1-\alpha)p_i, & m_i = 1, \\ 1, & m_i = 0, \end{cases} \qquad q_j = \prod_{t=1}^{j} \bar{s}_t, \tag{13}$$

$$w_i = \mathrm{sg}\left(\sum_{j=i}^{P} m_j q_j \right), \qquad \mathcal{L}_{\text{suffix}} = \frac{1}{N} \sum_{\text{blocks}} \sum_{i=1}^{P} m_i w_i \ell_i. \tag{14}$$

Here N is the number of training sequences used for normalization. The stop-gradient operator sg holds probabilities and weights fixed during backpropagation; invalid positions contribute zero.

For the causal prefix objective, the auxiliary queries share one draft forward pass and common prefix states. Each content node attends only to itself and earlier content in the same block; query i attends to preceding content and its own MASK, excluding y_i, future content, other queries, and other blocks. This attention graph equals independent query evaluation and adds no inference parameters. Causality applies only to prefix-CE; suffix training and packed inference remain bidirectional. The auxiliary single-MASK query differs from the full-MASK-suffix query defining equation 2.

B GSM8K and HumanEval Ablation Results

Tables 3 and 4 report speedup and acceptance length for the depth/pass-budget and prefix-loss studies on 128 GSM8K prompts and 164 HumanEval prompts. The HumanEval evaluation changes only

the benchmark: each study retains its checkpoints, HF inference path, attention backend and bidirectional attention, pass budgets, warmup, and timing protocol from the corresponding GSM8K evaluation. The multi-pass entries use first-MASK certification, and one-pass decoding records are reused. Both target-only and speculative TPOT use the returned token count N, following Appendix D; normalization leaves the measured intervals and AAL unchanged. The tables report dataset-specific results. Figure 4 presents the GSM8K curves. Appendix C reports the training-query and branch-count controls on GSM8K.

The additional Branch-Jacobi series in Figure 4 (left) shares its GSM8K measurements with the $\lambda = 0.2$ rows of Table 4. It is distinct from the three-layer depth-study rows, which use a different DFlash-loss checkpoint and refinement implementation (Appendix D).

On both datasets, additional passes increase AAL at every tested depth, with depth-dependent gains in speedup. At $F = 4$, all tested nonzero prefix-loss weights yield higher AAL than $\lambda = 0$ on both datasets. These studies characterize the trade-off between draft capacity, refinement cost, and acceptance: the preferred depth and loss weight depend on the metric, dataset, and pass budget. Among the configurations reported in Table 1, the three-layer drafter with three passes achieves the highest measured seven-benchmark geometric-mean speedup in all four model–temperature settings.

Table 3: Draft-depth and pass-budget ablation on GSM8K (128 prompts) and HumanEval (164 prompts), using Qwen3-4B at $T = 0$ and the same 100K-trained checkpoints. $F = 1$ records are reused; $F = 2$–5 use first-MASK certification. Both target-only and speculative TPOT use the common denominator N of returned tokens.

Draft layers (L)	Passes (F)	GSM8K		HumanEval	
		Speedup	AAL (τ)	Speedup	AAL (τ)
1	1	2.37×	2.68	2.21×	2.47
1	2	2.63×	3.11	2.59×	2.85
1	3	2.79×	3.26	2.61×	2.98
1	4	2.70×	3.30	2.63×	3.02
1	5	2.69×	3.32	2.52×	3.04
2	1	3.43×	3.99	3.21×	3.74
2	2	4.24×	5.09	3.99×	4.78
2	3	4.57×	5.59	4.12×	5.25
2	4	4.52×	5.82	4.12×	5.45
2	5	4.60×	5.90	4.13×	5.51
3	1	3.72×	4.46	3.55×	4.26
3	2	4.57×	5.66	4.48×	5.35
3	3	4.78×	6.16	4.65×	5.76
3	4	4.67×	6.29	4.45×	5.92
3	5	4.69×	6.36	4.41×	5.96
4	1	3.88×	4.78	3.65×	4.51
4	2	4.47×	5.95	4.47×	5.55
4	3	4.80×	6.33	4.47×	5.89
4	4	4.77×	6.50	4.53×	6.01
4	5	4.71×	6.54	4.29×	6.05
5	1	3.86×	4.90	3.66×	4.65
5	2	4.39×	5.98	4.20×	5.66
5	3	4.65×	6.41	4.31×	6.03
5	4	4.54×	6.56	4.26×	6.15
5	5	4.48×	6.60	4.20×	6.20

Table 4: Prefix-loss ablation on GSM8K (128 prompts) and HumanEval (164 prompts), using Qwen3-4B at $T = 0$ and the same 100K-trained checkpoints. $F = 1$ records are reused; $F = 2$–5 use first-MASK certification. Both target-only and speculative TPOT use the common denominator N of returned tokens. The five-layer root-only reference uses existing one-pass records; its GSM8K entry shares the $F = 1$ records with Table 5.

λ	Passes (F)	GSM8K		HumanEval	
		Speedup	AAL (τ)	Speedup	AAL (τ)
0.0	1	4.11×	4.93	3.86×	4.65
0.0	2	5.40×	6.07	4.32×	5.71
0.0	3	5.49×	6.52	4.61×	6.09
0.0	4	5.70×	6.66	4.36×	6.21
0.0	5	5.60×	6.70	4.57×	6.24
0.2	1	4.14×	4.99	3.87×	4.67
0.2	2	4.79×	6.21	4.14×	5.82
0.2	3	5.14×	6.67	4.53×	6.24
0.2	4	5.02×	6.82	4.38×	6.40
0.2	5	4.77×	6.85	4.43×	6.45
0.4	1	4.09×	4.92	3.84×	4.64
0.4	2	4.78×	6.20	4.45×	5.78
0.4	3	5.03×	6.68	4.72×	6.21
0.4	4	5.03×	6.84	4.73×	6.38
0.4	5	4.94×	6.90	4.59×	6.45
0.6	1	4.06×	4.90	3.83×	4.59
0.6	2	4.82×	6.17	4.41×	5.78
0.6	3	5.09×	6.72	4.77×	6.23
0.6	4	5.00×	6.94	4.84×	6.43
0.6	5	5.04×	7.00	4.72×	6.48
0.8	1	4.05×	4.85	3.81×	4.55
0.8	2	4.67×	6.09	4.63×	5.75
0.8	3	5.09×	6.66	4.98×	6.24
0.8	4	5.01×	6.84	4.98×	6.41
0.8	5	4.95×	6.91	4.82×	6.47
1.0	1	4.05×	4.83	3.77×	4.51
1.0	2	5.25×	6.05	4.94×	5.70
1.0	3	5.55×	6.60	5.08×	6.22
1.0	4	5.44×	6.80	5.21×	6.42
1.0	5	5.38×	6.88	5.00×	6.47
Root-only (5 layers)	1	4.15×	5.25	3.91×	4.98

C Controlled Ablations of Training and Inference

Training for moving-anchor refinement. We compare root-only and moving-anchor five-layer drafters under matched training conditions, changing only the training query geometry. Table 5 reports the GSM8K measurements. The moving-anchor checkpoint is the same five-layer checkpoint used in the depth study; these entries are shared. From $F = 1$ to $F = 4$, root-only AAL falls from 5.25 to 4.02, whereas moving-anchor AAL rises from 4.90 to 6.56. Their corresponding speedups change from $4.15\times$ to $2.94\times$ and from $3.86\times$ to $4.54\times$. These matched results support training on revealed prefixes for multi-pass refinement.

Active branch count in Jacobi decoding. Using the same drafter and first-MASK certification principle, we compare one active refinement branch ($K = 1$) with the packed branch family ($K = 14$). Here K labels the branch configuration: $K = 14$ denotes the full family at the first refinement, while the number of logically active branches contracts as certification advances. At $F = 1$, both configurations execute the same root-only drafting procedure and provide separate timing measurements of this shared procedure. The single-branch control uses a dedicated update: it queries only the current certified anchor $s = c$, returns its complete candidate, and certifies exactly the next token. The packed variant uses all live anchors to identify the common-prefix boundary and select a complete branch. Table 6 reports this comparison on GSM8K. The $K = 1$ decoding records are reused, and the packed-family results are shared with the three-layer depth study. Each configuration's speedup uses its independently measured AR reference. At each budget $F = 2$–5, packed branches achieve higher AAL and measured speedup. At $F = 4$, AAL increases from 5.44 to 6.29 and speedup from $4.15\times$ to $4.67\times$, indicating that access to multiple anchors compensates for the larger draft batch on these GSM8K prompts.

Table 5: Root-only and moving-anchor five-layer drafters on GSM8K with Qwen3-4B at $T = 0$, trained on the same 100K subset with matched architecture and loss.

Training queries	Passes (F)	GSM8K	
		Speedup	AAL (τ)
Root-only (5 layers)	1	4.15×	5.25
	2	3.12×	4.04
	3	3.01×	4.03
	4	2.94×	4.02
	5	2.81×	4.02
Moving-anchor (5 layers)	1	3.86×	4.90
	2	4.39×	5.98
	3	4.65×	6.41
	4	4.54×	6.56
	5	4.48×	6.60

Table 6: One active refinement branch ($K = 1$) and the packed branch family ($K = 14$) on GSM8K with Qwen3-4B at $T = 0$, sharing a three-layer drafter trained on the 100K subset. Branch configurations follow Appendix C.

Branch configuration (K)	Passes (F)	GSM8K	
		Speedup	AAL (τ)
1	1	3.71×	4.46
	2	3.92×	4.88
	3	4.06×	5.18
	4	4.15×	5.44
	5	4.20×	5.68
14	1	3.72×	4.46
	2	4.57×	5.66
	3	4.78×	6.16
	4	4.67×	6.29
	5	4.69×	6.36

D Experimental Details

All evaluations use Qwen3-4B or Qwen3-8B with thinking mode disabled and at most 2048 generated tokens; DFlash and Branch-Jacobi use block size 16. Branch-Jacobi uses three-layer DFlash-style drafters trained from random initialization for six epochs while the target remains frozen. The 2,480,285-example mixture combines NVIDIA Nemotron Post-Training Dataset v2 (Nathawani et al., 2025), CodeAlpaca-20k (Chaudhary, 2023), and Step-3.5-Flash-SFT (Huang et al., 2026a); responses are regenerated by the corresponding Qwen3 target. Truncation to 3072 tokens and removal of examples with fewer than 32 supervised tokens leave 2,327,981 training examples for 4B and 2,326,917 for 8B. Training uses AdamW with a cosine schedule, learning rate 6×10^{-4}, 4% warmup, per-GPU batch size 2, two gradient-accumulation steps, and seed 42. The suffix loss uses D-PACE continuation weighting with $\alpha = 0.5$ (Wu et al., 2026), and the causal prefix loss uses $\lambda = 0.2$. Main results use the sixth-epoch checkpoints. The controlled studies use a shared 100K subset. The DFlash reference in Figure 4 and Table 4 is our five-layer root-only drafter trained on this subset. Its GSM8K entry shares the existing $F = 1$ decoding records with Table 5; the HumanEval entry reuses the same checkpoint's HumanEval $F = 1$ records. This control differs from the public DFlash checkpoints used in the main table. Its training subset matches the three-layer Branch-Jacobi series in the left panel, but its drafter depth and training objective differ. The root-only/moving-anchor comparison matches the five-layer architecture, initialization seed, optimizer schedule, batch size, six-epoch budget, and DFlash loss while changing the training query geometry. The $K = 1/K = 14$ comparison uses one shared three-layer checkpoint and first-MASK certification, restricted to the available branches as described in Appendix C.

The main-table DFlash baselines use the public five-layer `z-lab/Qwen3-4B-DFlash-b16` and `z-lab/Qwen3-8B-DFlash-b16` checkpoints; EAGLE-3 uses the public `AngelSlim/Qwen3-4B_eagle3` and `AngelSlim/Qwen3-8B_eagle3` checkpoints with tree sizes 16 and 60. All methods run through Hugging Face/PyTorch on one H100 in bfloat16 with batch size and concurrency one. The 100K depth and controlled studies use batched ragged FlashInfer refinement; the prefix-loss sweep retains its per-row FlashInfer CUDA-graph implementation. Both preserve bidirectional attention within each branch, and timing comparisons between these sweeps also reflect this implementation difference. For Branch-Jacobi, the target and initial drafter pass use FlashAttention-3; packed refinement uses FlashInfer cascade attention. The first refinement has 14 packed rows and later refinements have 13; these physical shapes stay fixed as the logical active set contracts. All F draft forwards execute even when certification finishes earlier; further passes then preserve the proposal. At $T = 1$, DFlash uses full-softmax proposals, Branch-Jacobi uses delta proposals with exact residual sampling, and EAGLE-3 uses its public posterior sampler. All use the same untruncated target softmax at temperature 1. Fixed evaluation subsets contain 30 AIME25, 164 HumanEval, 80 two-turn MT-Bench, and 128 prompts from each remaining benchmark. AIME25 uses the MathArena distribution of the 2025 AIME problems.[1]

[1] `https://huggingface.co/datasets/MathArena/aime_2025`, `train` split.

The serial causal Draft-AR control in Figure 2 uses the same final Qwen3-4B checkpoint and MATH-500 cohort as our refinement curves. It generates each proposal token with a two-position $[\text{carrier}, \text{MASK}]$ query under causal attention, retaining previous carriers in the draft KV cache. Its AAL provides the empirical reference for gap recovery. The recursive drafter-AR path D in Eq. equation 2 is defined by full-MASK suffix queries with bidirectional branch attention and serves as the reference for the certification analysis.

The starred MTP, DFlash, DSpark, and DFlash 2 curves in Figure 2 are adapted from Figure 5 of the DFlash 2 technical blog (Inco AI, 2026). That source evaluates Qwen3.5-4B on MATH-500 with thinking enabled, $T = 1$, top-$p = 0.95$, top-$k = 20$, and presence penalty 1.5. Each external curve is shifted by its first-position acceptance rate to show the position-wise trend. Our curves use Qwen3-4B with thinking disabled and $T = 0$; the external curves illustrate qualitative position-wise trends under their respective evaluation settings.

Figure 1 combines separately collected CUDA-event stage profiles and decode-stage benefit measurements. Branch-Jacobi benefit measurements use the same decoding records as Table 7; profiling runs use the same checkpoint and prompts. Measurements follow warm-up generations. Target-only timing excludes prompt prefill and generation of the first token. For Branch-Jacobi and DFlash, speculative timing additionally excludes the first draft forward of each response, which primes the draft cache. It includes all refinement passes, subsequent draft forwards, target verification, sampling, and cache updates. Both target-only and speculative TPOT divide the measured interval by all N returned tokens, including the first token. Reported speedups quantify the measured decode interval under these timing boundaries and the common N-token normalization.

Tables 7–9 use the final Qwen3-4B checkpoint on 128 MATH-500 responses at $T = 0$. Decode-stage TPOT and AAL come from decoding runs; stage costs come from CUDA-event profiles. The runtime statistics use the main evaluation records; stage costs and certification traces come from separate diagnostic runs whose outputs and acceptance sequences match those records. The $F = 1$ decoding records are reused, with speedups reported under the same normalization. The 95% runtime intervals use paired-response resampling to quantify variation across prompts. Across seven benchmarks, a paired prompt bootstrap with 10,000 within-benchmark resamples, keeping MT-Bench turns together, gives 95% intervals for the $F = 4$ speedup gain over DFlash of [22.3, 24.4]% and [22.3, 26.0]% for 4B at $T = 0, 1$, and [26.7, 28.9]% and [23.2, 26.8]% for 8B. At $T = 1$, pairing is by original prompt; sampled outputs and second-turn contexts can differ between methods.

Table 7: Decode-stage latency and acceptance length on Qwen3-4B/MATH-500.

Method	TPOT (ms)	95% CI	AAL
B-Jacobi $F = 1$	4.312	[4.154, 4.477]	7.650
B-Jacobi $F = 2$	3.672	[3.542, 3.805]	9.268
B-Jacobi $F = 3$	3.557	[3.435, 3.684]	9.787
B-Jacobi $F = 4$	3.529	[3.412, 3.652]	9.936
B-Jacobi $F = 5$	3.750	[3.624, 3.884]	9.972

Table 7 evaluates the latency–acceptance trade-off and uncertainty in adjacent pass budgets. The paired $F = 4$ minus $F = 3$ TPOT difference is -0.0277 ms, with a 95% interval of $[-0.0567, 0.0002]$ ms. This interval includes zero. Moving from $F = 4$ to $F = 5$ increases TPOT by 0.2210 ms (95% interval [0.1864, 0.2566] ms). Table 8 isolates the per-cycle cost components: each extra refinement adds draft work that must be offset by greater target acceptance. Stage profiles report cycle-weighted model-forward CUDA-event times, with each response's first decoding cycle excluded. Certification tracing and its synchronization run separately. These profiles attribute model-forward costs, complementing the decode-stage TPOT measurements that also include sampling and cache updates.

For each verification cycle, c is the final certified-prefix length and a is the number of draft tokens accepted by the target before the bonus token. Table 9 complements the conditional curves in Figure 4 with cycle-weighted means and rank correlations. The correlations are positive but decrease with F, while certified length increases more than accepted length. These statistics weight verification cycles equally; the main-table AAL averages each response's mean acceptance length, including the bonus token, across responses. Every diagnostic response matches the main evaluation's output

Table 8: Cycle-weighted draft and target stage costs.

F	Initial draft (ms)	Refinement/pass (ms)	Target verify (ms)	Packed rows
1	3.051	–	26.289	–
2	3.228	1.319	27.623	14
3	3.224	1.248	26.719	14/13
4	3.229	1.180	26.494	14/13
5	3.236	1.179	25.829	14/13

and acceptance sequence. An independent check of certification against the recursive full-MASK drafter-AR path covers the first two cycles of one pilot response for each $F = 2$–5, totaling eight cycles.

Table 9: Certified-prefix length and target acceptance on Qwen3-4B/MATH-500.

F	Mean c	Mean a	Spearman $\rho(c, a)$
1	1.000	6.465	–
2	8.281	8.033	0.702
3	11.434	8.542	0.615
4	13.200	8.680	0.518
5	14.124	8.722	0.417