

# Dual-Draft Routing: Two Trees Enter, One Tree Gets Verified

February 25, 2026

**One sentence.** At every decode step, both draft heads propose speculative trees; we score each tree by a confidence statistic; we verify only the better-scoring tree with the base model; then both heads advance using the same accepted hidden states so they remain interchangeable at the next step.

## 1 Cast of Models

We generate tokens for a prompt  $x$ . Let  $y_{1:t}$  be the already-accepted continuation at time  $t$ .

- **Verifier (base LLM).** A distribution  $p_V(\cdot \mid x, y_{1:t})$  that defines correctness for acceptance.
- **Two draft heads.** For  $h \in \{1, 2\}$ , a draft distribution  $p_{D_h}(\cdot \mid x, y_{1:t})$  used only to propose candidates quickly.

## 2 The Root Token Trick

Each speculative step begins by producing a *root sample token*  $s_{t+1}$  from the verifier under the chosen decoding policy (greedy or sampling). Every draft tree from every head is rooted at this same token. The intent is that the verifier’s next-token decision is always represented in the tree, and bookkeeping stays consistent across steps.

## 3 What a Draft Head Produces

Fix  $t$  and the current prefix  $y_{1:t}$ . Head  $h$  produces a rooted candidate tree  $\mathcal{T}_h$  whose nodes are tokens extending the root  $s_{t+1}$ . The tree implicitly defines a set of candidate paths (root-to-leaf sequences)  $\mathcal{P}_h$ . For each path  $p \in \mathcal{P}_h$  we write its tokens as

$$u_{1:\ell(p)}^{(h,p)} = \left( u_1^{(h,p)}, \dots, u_{\ell(p)}^{(h,p)} \right).$$

**Tree budget.** The tree is constructed by repeatedly taking top- $k$  expansions up to a depth limit, then keeping only a fixed number of draft nodes (a token budget). So the head is not returning *all* paths, only the best-scoring ones under its own draft scores.

## 4 Draft Confidence as “Exponentiated Cumulative Log-Score”

Every retained draft node  $i$  in  $\mathcal{T}_h$  lies on some prefix of some candidate path. Let  $d(i)$  be its depth from the root (excluding the root itself). Define a cumulative draft log-score

$$s_{h,i} = \sum_{j=1}^{d(i)} \log p_{D_h}(u_j \mid x, y_{1:t}, s_{t+1}, u_{1:j-1}).$$

The confidence attached to node  $i$  is simply

$$c_{h,i} = \exp(s_{h,i}).$$

This maps cumulative log-probability back into probability space.

## 5 A Single Scalar Per Head

The router needs one number per head, per step. We use the mean confidence over the retained nodes:

$$\mu_h = \frac{1}{N_h} \sum_{i=1}^{N_h} c_{h,i},$$

where  $N_h$  is the number of retained draft nodes in  $\mathcal{T}_h$  at that step.

**Routing rule.** Choose the head with larger mean confidence:

$$h^* = \arg \max_{h \in \{1,2\}} \mu_h.$$

Verify only  $\mathcal{T}_{h^*}$ .

## 6 Verifier Acceptance (What “Verification” Actually Means)

The verifier scores the chosen tree  $\mathcal{T}_{h^*}$  in a single batched forward pass, producing logits for every node position along every candidate path. Then the algorithm selects:

- a best candidate path  $p^* \in \mathcal{P}_{h^*}$ , and
- an accepted prefix length  $a \geq 0$  along that path.

### 6.1 Greedy acceptance (temperature = 0)

Let  $\hat{z}_{t+j}$  denote the verifier argmax token at position  $t+j$  under greedy decoding. For a candidate path  $p$ , define the match indicator at depth  $j$  by

$$m_j^{(p)} = \mathbf{1} \left[ u_j^{(h^*, p)} = \hat{z}_{t+j} \right].$$

The accepted length for that path is the longest all-ones prefix:

$$a^{(p)} = \max \left\{ r : \prod_{j=1}^r m_j^{(p)} = 1 \right\}.$$

The verifier chooses  $p^*$  that maximizes  $a^{(p)}$  and sets  $a = a^{(p^*)}$ .

### 6.2 Sampling acceptance (temperature > 0)

Under sampling, verifier logits are transformed by a decoding operator  $\mathcal{G}$  (temperature, top- $p$ , top- $k$ ). Acceptance proceeds sequentially along the candidate path: at each depth  $j$ , the candidate token is accepted with probability proportional to the verifier probability mass assigned to that token (under the current transformed distribution), and a rejection triggers a renormalization step that removes the rejected token and samples the next token from the remaining mass. The accepted length  $a$  is the number of draft tokens accepted before the first rejection along the selected path.

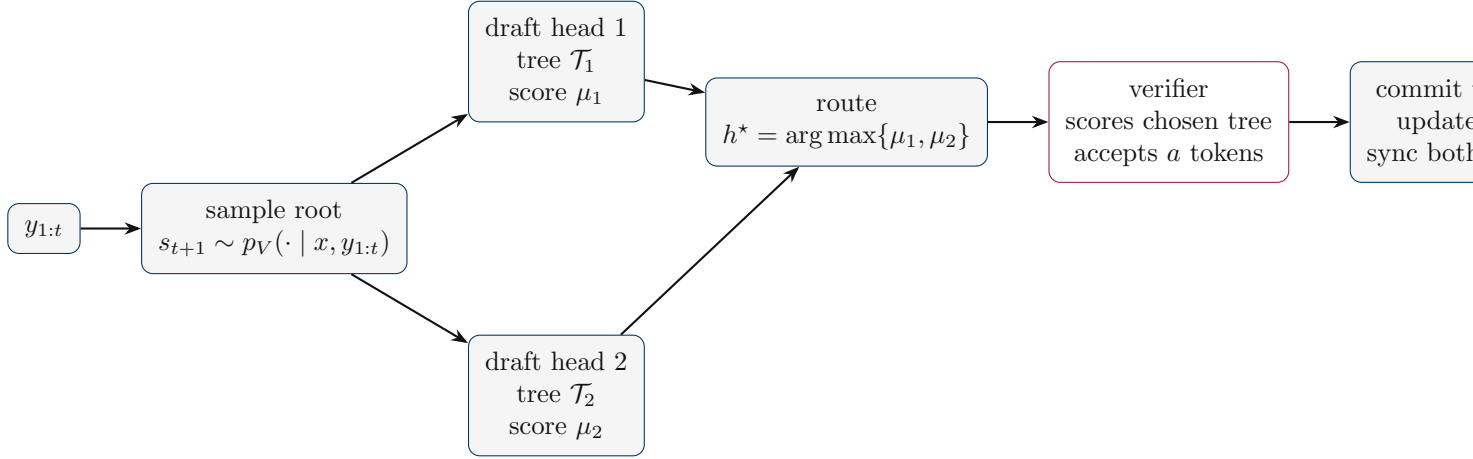
## 7 Commit and Synchronize

After verification, we append the accepted tokens to the continuation:

$$y_{1:t} \leftarrow (y_{1:t}, s_{t+1}, u_{1:a}^{(h^*, p^*)}).$$

**Interchangeability lemma (why routing is safe).** Both draft heads are advanced using the same accepted hidden states produced by the verifier for the appended tokens. Therefore, at the next step both heads are conditioned on the same prefix and can again propose a tree for routing.

## 8 The Whole Dance in One Picture



## 9 Efficiency Lens: $\tau$

Let “one step” mean one verifier pass over one draft tree. If step  $k$  accepts  $A_k$  tokens into the continuation (including the root token bookkeeping), then

$$\tau = \frac{\sum_k A_k}{\text{\#steps}}.$$

Routing changes which draft tree is verified at each step. Verifier computation per step is fixed by the tree budget, and the additional routing overhead is the cost of generating both draft trees and computing  $\mu_1$  and  $\mu_2$ .

## 10 Alternative: Verify Both Trees

Instead of choosing one head, one can merge the two draft trees into a single combined candidate set and verify the union in one verifier pass. Step-level routing verifies one tree; merged-tree verification verifies both. The tradeoff is: more draft-side compute for routing versus more candidate coverage for merged verification.