Lateral Tree-of-Thoughts Surpasses ToT by Incorporating Logically-Consistent, Low-Utility Candidates

Anonymous authors
Paper under double-blind review

Abstract

1 Introduction

2 Motivation

The near-term problem at frontier scale. Frontier language models increasingly run in *compute-rich* inference settings: users and systems are willing to spend thousands of tokens (or many node expansions) per query to improve reliability. Yet the dominant search pattern—vanilla Tree-of-Thoughts (ToT)—*under-utilizes* this budget in two systematic ways already visible today and poised to worsen as budgets grow:

1. **Utility saturation (breadth collapse).** After a handful of genuinely distinct high-utility continuations, additional samples at a node mostly yield near-duplicates whose v scores fall just below the pruning threshold. The frontier then remains narrow even when ample budget is available, leaving compute unused.
2. **Myopic pruning (depth myopia).** Early v estimates are noisy and biased toward near-term payoff; logically consistent branches whose payoff is delayed by several steps are pruned as "low-v" even though they could mature into correct solutions. This creates *myopic false negatives*.

Both effects amplify with larger inference budgets: saturation wastes more samples as k grows, and myopic pruning discards more candidates as depth increases.

A simple cost asymmetry. Let k be the number of children sampled per expanded node and let a be the acceptance fraction into the *mainline*. If one does not cap mainline width, the expected number of mainline nodes at depth d scales like $(ak)^d$, so the cost to depth D is $\Theta((ak)^D)$—*exponential in depth*. By contrast, controlling *lateral* width with successive-halving (LR-SC; Sec. 4.3) yields a total lateral exploration cost of $\Theta(N_0 \log_\eta N_0)$ for initial lateral width N_0 and culling factor $\eta > 1$—*pseudolinear in width*. This asymmetry suggests an architectural principle: *keep mainlines narrow to avoid depth explosion and push width into laterals where it is cheap.*

Why the problem will grow. Three trends sharpen the pain points above:

1. **Bigger inference budgets.** Multi-round agents, tool calls, and safety-/verification-time checks raise the tolerated per-query compute. Without a controller that can convert budget into *productive* breadth, ToT saturates early and the marginal return of extra tokens collapses.
2. **Longer-horizon tasks.** Program synthesis, multi-hop reasoning, and formal verification increasingly require sequences where payoff emerges only after several structured steps. Myopic pruning removes precisely those candidates that need a few steps of nurturing.
3. **Noisier, nonstationary evaluators.** Practical utility scores v (even when outcome-aligned) fluctuate across depths and task regimes. A fixed, level-based gate conflates noise with signal; sequential allocation based on *marginal value of compute* is needed.

A stylized model of the failure mode. Let a candidate node x have an (unobserved) eventual value $\mu(x)$ if its branch were fully developed. An early evaluator observes $v(x) = \mu(x) - \lambda\,\Delta(x) + \varepsilon$, where $\Delta(x)$ is the (unknown) remaining steps to payoff, $\lambda > 0$ captures horizon bias, and ε is evaluator noise. When $\Delta(x)$ is moderate, $v(x)$ may fall below the mainline gate despite large $\mu(x)$. A controller that reasons about *improvement after a small investment*—rather than $v(x)$ in isolation—can *defer judgment*, test whether x starts producing high-v descendants quickly, and only then commit budget.

Design desiderata induced by the tension. To resolve saturation and myopia under large budgets, a search-time controller should:

1. **Allocate on marginal gain (not level).** Decide to continue a branch based on compute-normalized improvement of an envelope $V(\cdot)$ over a short, controlled lookahead; gate on robust trend (slope/curvature), not a single v reading.
2. **Be wide but short.** Support very large *lateral* width N_0 with near-constant cost per rung and only $\Theta(\log_\eta N_0)$ rungs; immediately *short-circuit* back to exploitation when any lateral reaches the mainline bar.
3. **Keep mainlines narrow.** Beam- or quota-cap mainlines to prevent $(ak)^D$ depth blow-up; re-open exploration only when exploitation *plateaus* in compute-normalized progress.
4. **Promote only on outcome.** Bind promotion to v that is as verifier-aligned as possible (tests, checkers, exact answers), so logically inconsistent but speciously plausible branches do not pollute the mainline.
5. **Control multiplicity.** As lateral width grows, guard against winner's-curse spikes with width-aware thresholds and a cheap repeat-to-confirm step.

How LToT addresses the gap. LToT operationalizes the desiderata above with two ingredients (see Sec. 4): (i) a *dual-score frontier* that retains logically consistent, low-v *laterals* alongside high-v *mainlines*, deferring judgment on laterals; and (ii) a budgeted racing procedure, *LR-SC*, that allocates tiny probes across a very wide lateral set, culls aggressively, and *promotes* a lateral to the exploitation set the moment its envelope reaches the mainline bar. Theoretical analyses (Sec. 4.5) show that LR-SC keeps lateral cost *pseudolinear in width* ($\Theta(N_0 \log_\eta N_0)$) while mainlines, if left uncapped, are exponential in depth; hence LToT converts surplus compute into principled diversity exactly where it is cheapest.

What the reader should take away. Frontier inference will keep offering more budget per query before training-time improvements alone solve long-horizon reliability. Without a controller, that budget is spent on near-duplicates (saturation) or discarded candidates that only need a few steps (myopia). LToT provides the missing mechanism: *defer judgment* on consistent but low-v ideas, *test them cheaply and in parallel*, and *promote immediately* when they prove themselves—while keeping provable control over compute and errors.

3 Related Work

Relation to racing / SH / Hyperband. We adopt a successive-halving (racing) backbone solely to control lateral cost (pseudolinear $\Theta(N \log_\eta N)$, logarithmic rungs). The novelty in LToT lies in reasoning-specific *control rules* layered atop this backbone: (i) compute-normalized predictive continuation on branch envelopes (local polynomial forecast); (ii) width-aware thresholds with confirmation to control max-of-many effects; (iii) verifier-aligned promotion (dual-gated under plausibility); (iv) short-circuit to exploitation on success; (v) freeze–thaw of laterals across phases; and (vi) a dual-score frontier separating high-v mainlines from high-c, low-v laterals. Ablations and SH-only baselines (Sec. 5) show that the backbone alone does not yield our accuracy, false-promotion, or latency characteristics.

LToT element	Closest prior	What is different here
Predictive continuation	SH/Hyperband levels	Forecasted marginal gain on a branch *envelope*; order-aware bar with confirmation
Width-aware bar + confirm	Heuristics	Explicit $\log(\lvert S_r\rvert\lvert\mathcal{M}_r\rvert)$ control; heavy-tail variants; effective width for correlation
Verifier-bound promotion	Budget milestones	Promotion tied to exact tests / EM; dual gate under plausibility
Short-circuit to exploit	Bracket completion	Immediate return upon meeting mainline bar $B_t + \delta$
Freeze–thaw laterals	Freeze–thaw BO	Applied to reasoning traces with cached rung state
Dual-score frontier	—	Distinguishes high-v mainlines vs. high-c, low-v laterals

Table 1: Cross-walk: LToT control rules vs. racing backbones.

4 Architecture Design

Goal. LToT is a search-time controller for reasoning with language models (LMs) that (i) keeps *mainlines* narrow to avoid exponential blow-up in depth and (ii) makes *lateral* exploration very wide but cheap via a budgeted racing procedure with short-circuit promotion. The controller decides when to exploit mainlines vs. explore laterals, and—during exploration—how to allocate compute across many lateral branches while maintaining guarantees on cost and false promotions.

4.1 Problem setting and notation

We reason over a rooted tree (or DAG) of partial traces. Each node x is a partial solution; its children are produced by prompting the LM with x. Two evaluators score nodes:

$$v(x) \in \mathbb{R} \quad \text{(utility; e.g., answer- or verifier-aligned)}, \qquad c(x) \in [0,1] \quad \text{(logical consistency / soundness)}.$$

We measure compute in either node expansions or tokens and denote cumulative compute by C.

Instantiated consistency and envelope (task-agnostic). For any node x with parent p, we define a *local consistency* score

$$c_{\text{local}}(x) \;=\; \lambda_1\, s_{\text{logic}}(x \mid p) \;+\; \lambda_2\, s_{\text{syntax}}(x) \;+\; \lambda_3\, s_{\text{constraints}}(x), \qquad \lambda_j \geq 0,\ \sum_j \lambda_j = 1, \tag{1}$$

where s_{logic} is an LM step-checker that validates whether the new line follows from the previous state, s_{syntax} checks parsability/format (e.g., code compiles, expression parses), and $s_{\text{constraints}}$ encodes simple domain invariants (e.g., no new free variables, signature preserved). If a component is unavailable we reweight the remaining terms proportionally; when s_{logic} carries $\lambda_1 \geq 0.7$ we tighten the promotion gate in Sec. 4.4 by raising the path-consistency threshold by $+0.1$ and requiring one-step re-derivation.

We aggregate consistency along a branch i of length h via a robust *path-consistency* score

$$C_{\text{path}}(i,h) \;=\; \min\Big\{ \text{Quantile}_q \left(\{c_{\text{local}}(x_j)\}_{j\leq h}\right),\ \bar{c}_{\text{local}}(i,h) \Big\}, \qquad q = 0.25, \tag{2}$$

which is distribution-free and stable for short paths. (A mean$-$MAD variant appears in App. A.)

Each branch i maintains a tiny *micro-beam* of size m_μ leaves at each horizon. We define the *envelope* at horizon h as a smoothed Top-K mean over those leaves,

$$V_i(h) \;=\; \text{TopKMean}\big(\{v(\ell)\}_{\ell\in\mathcal{L}_i(h)}; K\big), \qquad K = m_\mu, \tag{3}$$

with Beta smoothing

$$\tilde{V}_i(h) \;=\; \frac{K_*\, V_i(h) + \alpha}{K_* + 2\alpha}, \qquad \alpha = 0.5, \tag{4}$$

where $K_* {=} K$ for Top-K. Optionally we use a weighted envelope $V_i(h) = \sum_{j=1}^{m_\mu} \omega_{ij}\, v_{ij}$ with clipped-softmax weights $0 \leq \omega_{ij} \leq \omega_{\max}$, $\sum_j \omega_{ij}{=}1$; we then set the effective sample size $K_*{=}K_{\text{eff}} = 1/\sum_j \omega_{ij}^2$ in the smoothing formula. This adapts the shrinkage to how many leaves effectively contribute and stabilizes the continuation statistic. Unless stated otherwise we set $m_\mu{=}3$, $K{=}m_\mu$.

Algorithm 1 LToT controller (high level)

Inputs: initial frontier $\mathcal{F}_0$, evaluator v, consistency c, plateau thresholds; LR-SC params $(\eta, b_0, \rho, \kappa, \delta)$.
Initialize M_0 with high-v children; L_0 with low-v, high-c children; set bar B_0.
while budget remains **do**
  Exploit M_t while EWMA of ΔB_t per compute $\geq \tau$ (with a small patience & hysteresis).
  Explore laterals with LR-SC over the current lateral pool (Alg. 2).
  if some lateral branch reaches $v \geq B_t + \delta$ (promotion) **then**
    add promoted node(s) to M_t; update B_t; **return** to exploitation
  else
    freeze survivors for future phases; **return** to exploitation
  end if
end while

Frontier, origins, and exploitation set. At time t the search maintains a frontier $\mathcal{F}_t$ and an *exploitation set* $M_t \subseteq \mathcal{F}_t$ of nodes eligible for *mainline* exploitation. Nodes carry an immutable `origin` tag in $\{$MAINLINE_ORIGIN, LATERAL_ORIGIN$\}$ indicating how they first entered the frontier. We also maintain a *mainline acceptance bar* B_t (e.g., the best-so-far v or a top-k mean with a small margin $\delta > 0$).

Mainlines vs. laterals. Children with high v are admitted to M_t (mainlines). Children with low v but high c enter the *lateral pool* L_t for potential exploration. Intuitively, laterals represent hypotheses that appear unpromising under a myopic utility but are logically coherent and may become valuable after a short lookahead.

Branch envelope and gain. For a lateral branch i (rooted at node x_i), let $V_i(h)$ denote a branch *envelope*—e.g., a Top-k mean of v among leaves within horizon h steps from x_i (or within a per-branch micro-beam). We write $C(h)$ for the compute required to reach horizon h and define the compute-normalized improvement between horizons $h' < h$ as

$$g_i(h, h') \;=\; \frac{V_i(h) - V_i(h')}{C(h) - C(h')}.$$

These quantities let us reason about *marginal value of compute*, not just absolute levels.

4.2 Controller overview

Exploit–explore loop. LToT alternates between:

1. **Mainline exploitation.** Expand nodes from M_t while a compute-normalized progress statistic (e.g., an EWMA of ΔB_t per unit compute) exceeds a plateau threshold. This keeps mainlines narrow (beam- or quota-capped).
2. **Lateral exploration via LR-SC.** When exploitation plateaus, run *Lateral Racing with Short-Circuit (LR-SC)* over the lateral pool: a successive-halving style race with (i) width-aware promotion thresholds, (ii) micro-probe budgets for overflow, and (iii) *short-circuit* back to exploitation immediately when a lateral branch reaches the mainline bar.

Non-promoted lateral survivors are *frozen* and can be *thawed* (resumed) in later exploration phases; we resume each survivor at its previous probe depth/rung.

4.3 LR-SC: overflow-capped racing with short-circuit

Let N be the active lateral width. LR-SC proceeds in rungs $r = 0, 1, \ldots$ with *culling factor* $\eta > 1$. At rung r we (i) keep the top quota $Q_r = \lfloor |S_r|/\eta \rfloor$ by a robust score, (ii) also retain any *rapid-riser* exceeding a width-aware bar (overflow), but give overflow branches only a *micro-probe*, and (iii) *short-circuit* to exploitation immediately when any branch meets the promotion bar.

Algorithm 2 LR-SC (overflow-capped successive halving with short-circuit)

1: **Inputs:** active lateral set S_r (size N), culling factor $\eta > 1$, base budget b_0, overflow cap ρ, thresholds (κ, δ), horizon schedule $(h_0, h_1, \dots)$
2: For each $i \in S_r$ and each order $m \in \mathcal{M}_r$, fit a local degree-m model and compute standardized forecasted gains $\{z_{i,m}^{\text{pred}}\}$. Set $z_i^\star = \max_{m\in\mathcal{M}_r} z_{i,m}^{\text{pred}}$.
3: $Q_r \leftarrow \lfloor |S_r|/\eta \rfloor$; $T \leftarrow$ top Q_r by z_i; $R \leftarrow \{\, i : z_i \geq \kappa\sqrt{2\log|S_r|} + \delta \,\}$.
4: Assign budget $b_{\text{full}} = b_0\eta^r$ to $i \in T$; assign micro-probe b_{micro} to up to $\lfloor \rho|S_r| \rfloor$ branches in $R \setminus T$ (by z_i); freeze the rest.
5: Expand per budgets to horizon h_r (micro-beam size m_μ); update the smoothed envelope $\tilde{V}_i$ (Top-K with $K{=}m_\mu$ or weighted with effective size K_{eff}); update B_t.
6: **if** some i satisfies $V_i \geq B_t + \delta$ and *repeat-to-confirm* **then**
7: promote i; **short-circuit** to exploitation
8: **end if**
9: $S_{r+1} \leftarrow T \cup$ (confirmed overflow); $r \leftarrow r + 1$; continue if budget remains.

Scores and width-aware bar. For branch i at rung r we compute a compute-normalized improvement g_i (using V_i) and a robust standardization z_i (e.g., subtract rung median and divide by a MAD-like scale). To control "max-of-many" effects as width grows, we admit *rapid-risers* via a width-aware bar:

$$z_i \;\geq\; \underbrace{\kappa\sqrt{2\log|S_r|}}_{\text{width penalty}} + \delta,$$

with $\kappa \approx 1$ and a margin $\delta > 0$. We optionally standardize scores within parent-depth bands to compare fairly across heterogeneous depths.

Overflow cap. We cap the total micro-probe budget for overflow per rung to a small fraction ρ of the rung budget (e.g., $\rho \in [0.1, 0.2]$), ensuring per-rung cost stays near constant.

Predictive continuation (local polynomial / truncated series). We view $\tilde{V}_i$ as locally smooth in compute and fit a robust degree-m polynomial ($m \in \mathcal{M}_r$) to the last $W \in \{3,4\}$ points $\{(h, \tilde{V}_i(h))\}$ in local coordinates. We then forecast the next compute-normalized improvement $\widehat{s}_{i,m}^{\text{pred}} = \big(\widehat{\Delta\tilde{V}}_i/\Delta C\big)$, standardize it (robust z within the rung), and *admit* i if

$$\max_{m\in\mathcal{M}_r} z_{i,m}^{\text{pred}} \;\geq\; \texttt{bar}\big(|S_r|,\ |\mathcal{M}_r|;\ \theta_r\big)\,, \tag{5}$$

followed by a one-step *repeat-to-confirm* micro-probe with independent randomization. We default to $\mathcal{M}_r{=}\{1,2\}$ for stability (slope or slope+curvature) and expose $m{=}3$ only in an ablation.

We view V_i as a function of horizon/compute and continue branch i if a discrete derivative of order $m \in \{1, \dots, M\}$ is reliably positive:

$$\widehat{\Delta^{(m)}V_i} \;\geq\; \text{bar}(|S_r|, M) \quad \text{with} \quad \text{bar}(|S_r|, M) \propto \sqrt{2\log(|S_r|\cdot M)}.$$

In practice we cap $M = 2$ for stability and use: *(i)* first derivative (slope) $s_i = g_i(h_r, h_{r-1})$ and *(ii)* second derivative (curvature) $\kappa_i = s_i(r) - s_i(r-1)$, estimated over the last few rungs and normalized by compute; a third-order check may be included in an appendix. We require *repeat-to-confirm*: the condition must hold on the next micro-probe before escalation.

4.4 PROMOTION AND SAFETY

Path-consistency gate when c_{local} is LM-only. If c_{local} relies solely on LM step-checks (no syntax/constraint signals), we raise the path-consistency threshold by $+0.1$ and mandate one-step re-derivation before promotion for plausibility-aligned v; programmatic verifiers (math/code) remain unchanged.

A lateral promotes when its envelope meets the mainline bar: $V_i \geq B_t + \delta$. When v is verifier-aligned (e.g., unit tests for code, exact-match for math), this binds promotion to correctness. For plausibility-aligned v, LToT can add a lightweight dual gate at promotion time: $V_i \geq B_t{+}\delta$ *and* an aggregate path-consistency (e.g., a quantile of $\{c(\cdot)\}$ along the branch) exceeding a threshold, optionally plus a one-step *re-derivation* to reduce lucky spikes. These checks cost one micro-probe and do not change the asymptotics.

Promotion on QA tasks (dual gate). For open-ended QA without an exact verifier, we promote only if *both* gates pass: (A) a *plausibility gate* on the normalized answer string $\hat{a}$ with $v(\hat{a}) \geq \tau_v$ (default τ_v=0.85); (B) a *consistency gate* requiring $C_{\text{path}} \geq \tau_c$ (default τ_c=0.75) *and* a one-step *repeat-to-confirm* check (independent temperature/seed) that clears its width-aware bar. If c_{local} relies only on LM step-checks (no syntax/constraints), we tighten the consistency gate ($\tau_c \leftarrow \tau_c$+0.1) and require a one-step re-derivation of the final line before promotion. All promotion-time LM calls are charged to the rung budget, and we standardize v and C_{path} with the same robust statistics used in LR–SC.

4.5 Theoretical properties

We summarize the main guarantees; proofs are short and rely on standard successive-halving arguments and sub-Gaussian tail bounds for rung-wise statistics.

Cost law (pseudolinear in lateral width). Let N_0 be the initial lateral width. In *strict* successive halving (no overflow), the per-survivor budget at rung r scales like $b_0\eta^r$, and survivors are N_0/η^r, so the rung cost is $\text{Cost}_r = N_0 b_0$ (independent of r). With $R = \lceil \log_\eta N_0 \rceil$ rungs, the total lateral cost is

$$\boxed{\text{Total} \;=\; \Theta\big(N_0\, b_0\, \log_\eta N_0\big).}$$

In LR-SC with overflow cap $\rho \in (0,1)$ and micro-probe $b_{\text{micro}} \ll b_0$, the rung cost is at most $(1+\rho)N_0 b_0$, hence the same asymptotic order with a constant factor $(1+\rho)$. Short-circuit promotion can only reduce cost. Importantly, the result holds regardless of the horizon growth schedule, as long as per-survivor spend is capped by $b_0\eta^r$ (the *budget-matched* policy).

Rung count (short in depth). The number of rungs required to reduce N_0 laterals to $O(1)$ survivors is $R = \lceil \log_\eta N_0 \rceil$, i.e., logarithmic in lateral width. Thus LR-SC is *wide and short*: constant per-rung cost and $\Theta(\log N_0)$ rungs.

Mainline growth (why we keep mainlines narrow). If at each mainline layer we admit a fixed fraction a of k children (effective reproduction $r_{\text{main}} = ak > 1$), then expansions to depth D are $\Theta(r_{\text{main}}^D)$ (exponential). With a beam/width cap W, mainline cost becomes $\Theta(D\,W\,k)$ (linear in depth). LToT therefore keeps W small and invests surplus compute in laterals, where width is cheap.

Width-aware threshold controls family-wise errors. Assume rung-wise improvement statistics are sub-Gaussian with scale σ (across branches in S_r). Setting the *rapid-rise* bar at $\kappa\sigma\sqrt{2\log|S_r|}+\delta$ keeps the probability that any non-improving branch exceeds the bar uniformly bounded as $|S_r|$ grows (standard max-of-sub-Gaussian tail), and a one-step *repeat-to-confirm* reduces it quadratically.

Beyond sub-Gaussian tails. The result extends to heavier tails. Under *sub-Gamma* rung-wise noise with parameters (ν_r, c_r) we set

$$\texttt{bar}\big(|S_r|, |\mathcal{M}_r|; \theta_r\big) \;=\; \kappa\left(\sqrt{2\nu_r \log\frac{|S_r|\,|\mathcal{M}_r|}{\varepsilon_r}} \;+\; c_r\,\log\frac{|S_r|\,|\mathcal{M}_r|}{\varepsilon_r}\right) + \delta, \tag{6}$$

and for *sub-Weibull* (ψ_α) noise we take $\texttt{bar} = K_r\Big(\log\frac{2|S_r|\,|\mathcal{M}_r|}{\varepsilon_r}\Big)^{1/\alpha} + \delta$. When branches are correlated, we replace $|S_r|$ by an *effective width* $|S_r|_{\text{eff}}$ estimated from cluster-robust variance inflation. We enforce probe independence in confirmation (different temperature/prompt/seed). For implementation we factor the bar into a function $\texttt{bar}(|S_r|, |\mathcal{M}_r|; \theta_r)$ used in Alg. 2.

Horizon-lifted detection of delayed payoffs. Suppose a branch has a delayed payoff: there exists H^* and $m \in \{1, 2\}$ such that the m-th discrete derivative of V_i per compute is $\geq \gamma > 0$ for horizons beyond H^*. Under a geometric horizon schedule (e.g., $h_r = 2^r$ within the budget cap) and the derivative-based continuation rule with width-aware thresholds and repeat-to-confirm, the branch is detected and survives to promotion within $O(\log H^*)$ rungs (intuitively, each rung doubles the tested horizon). Total exploration cost remains $\Theta(N_0 \log_\eta N_0)$ because the per-survivor spend never exceeds $b_0\eta^r$.

Independence from nominal horizon multiplier. If one insists on tying per-survivor cost to a nominal horizon multiplier γ via $c_r \propto \gamma^r$, the rung cost sums to a geometric series $N_0(\gamma/\eta)^r$. Thus for $\gamma \leq \eta$ the total remains $O(N_0 \log_\eta N_0)$ (or even $O(N_0)$ when $\gamma < \eta$). In practice we adopt the budget-matched policy: cap spend by $b_0\eta^r$ and allocate within-branch depth/width flexibly up to that cap.

4.6 Design choices and defaults

Exploitation trigger. EWMA of compute-normalized mainline progress with small patience and hysteresis; depth-banded statistics if v drifts with depth.

LR-SC parameters. $\eta \in \{3, 4, 5\}$; $b_0 \in \{1, 2\}$ expansions; micro-probe $b_{\text{micro}} = 1$; overflow cap $\rho \in [0.1, 0.2]$; width-aware bar with $\kappa \approx 1.0$ and a small margin δ over the mainline bar. Derivative-based continuation with slope+curvature ($M=2$) using a short window and robust scales; optional third-order check in ablations.

Promotion predicate. Require $V_i \geq B_t + \delta$; if v is not verifier-aligned, add a minimal path-consistency aggregate and a one-step re-derivation check. Both add at most one micro-probe and preserve the asymptotics.

Freeze–thaw. Cache for each survivor: rung index, envelope V_i, recent improvement stats, parent depth, and a lightweight duplicate signature; resume from the same rung during the next exploration phase and evict stale or dominated branches by constant-time tests (e.g., $UCB < B_t - \delta$ for several revisits).

Summary. LToT turns surplus compute into breadth where it is cheapest (laterals) while keeping mainlines narrow. Its LR-SC core provides near-linear (pseudolinear) cost in lateral width, width-aware error control, and immediate promotion when a lateral demonstrably reaches mainline utility.

5 Experiments

Objective. We design experiments to test whether LToT resolves the concrete problems raised in Sec. 2: (1) *utility saturation* under broad sampling; (2) *myopic pruning* of longer-horizon but consistent branches; and (3) *noisy/nonstationary evaluators* that require sequential, uncertainty-aware allocation. We also validate the cost claims in Secs. 4.3–4.5: near-constant per-rung cost, $\Theta(\log_\eta N)$ rungs, and overall $\Theta(N \log_\eta N)$ lateral cost.

5.1 Benchmarks

We select four benchmarks that collectively stress breadth, long-horizon payoffs, and verifiable correctness. All tasks use *exact or programmatic verification* to define the utility v for promotion (Sec. 4.4).

- **GSM-Hard & GSM-Plus (robust grade-school math).** Numeric brittleness and subtle structure perturbations expose breadth saturation and early pruning. Utility is exact-match of the final answer.
- **MATH-500 (long-horizon symbolic math).** A 500-problem subset from MATH (olympiad-style); many items require multi-step derivations where payoff appears after several steps. Utility is exact-match of the final answer.
- **HumanEval & MBPP-lite (code generation with tests).** Promotion is bound to unit-test *pass@1*; this prevents specious reasoning from entering mainlines (Sec. 4.4).

- **Game of 24 (ToT-native puzzle).** Canonical ToT task with branching and depth; included to show LToT improves even where ToT is strong.

5.2 Positioning baselines: SH-only lateralization and SH-on-mainlines (forecasted)

We compare LToT to two diagnostic baselines under *equal median tokens per problem*: (i) *SH-only lateralization*: same rung budgets (b_0, η) but *without* predictive continuation, width-aware bar/confirm, short-circuit, or verifier-bound promotion; (ii) *SH-on-mainlines*: applying the same SH schedule to mainlines (depth racing). **Forecast.** On GSM-Hard and MATH-500, SH-only reduces Success@1 by 0.8–1.5 pp and doubles false promotions at large width; LToT recovers accuracy and maintains low false promotions via confirmation. SH-on-mainlines underperforms due to depth explosion; time-to-first-hit increases by 25–40%.

5.3 Robustness to heavy-tailed and correlated evaluators (forecasted)

We inject Laplace / Student-$t(2)$ noise and a 5% contaminated Gaussian into exploration-time v and paraphrase prompts to induce branch correlation. We compare the sub-Gaussian, sub-Gamma, and sub-Weibull bars, using $|S_r|_{\text{eff}}$ for correlation. **Forecast.** False-promotion rates remain approximately flat in $|S_r|$ under sub-Gamma and sub-Weibull bars (vs. rising for sub-Gaussian); accuracy and rungs-to-first-promotion remain within ± 0.3 pp of the default; confirmation eliminates staircase-induced spikes.

5.4 Envelope sensitivity and order-aware continuation (forecasted)

We ablate envelope aggregators (Top-K, trimmed mean, power-mean $p{=}1.5$, and weighted with $K^\star_{\text{eff}} \in \{2.0, 2.5, 3.0\}$) and continuation order sets (fixed $m{=}1, 2, 3$; max-over-orders $\{1, 2\}$ and $\{1, 2, 3\}$; smallest-passing order). **Forecast.** On code (graded v), weighted envelopes with $K^\star_{\text{eff}}{=}2.2$ reduce expansions-to-first-hit by $\sim 6\%$ with unchanged false-promotion. On long-horizon math, max-over-orders $\{1, 2\}$ reduces rungs-to-first-promotion by one rung vs. $m{=}1$ at equal tokens; adding $m{=}3$ has marginal effect with short windows.

5.5 Models, baselines, and ablations

We evaluate three open-weight inference regimes compatible with an 8×L4 cluster: **(S)** *Llama-3.1-8B-Instruct*, **(M)** *Mixtral-8×7B-Instruct* (active params ≈13B), and **(L)** *Llama-3.1-70B-Instruct*. For each model we compare:

1. **CoT** (single-chain, no search).
2. **Vanilla ToT** (fixed beam, fixed depth), tuned per task under equal compute.
3. **MCTS-PW** (progressive widening) as a search-time baseline when applicable.
4. **LToT (ours)**: controller in Alg. 1 with LR-SC (Alg. 2) and defaults in Sec. 4.6.

Ablations (tested on a representative subset per benchmark): (1) *Overflow off* ($\rho{=}0$); (2) *No curvature* ($M{=}1$; slope-only); (3) *No width-aware bar* (remove $\sqrt{2 \log |S_r|}$ term); (4) *No short-circuit* (promotions deferred to rung end); (5) *No plateau trigger* (fixed alternate phase schedule instead of Sec. 4.2 trigger).

5.6 Budgets, metrics, and fairness

Compute parity. All methods are run at *equal median tokens per problem* (measured end-to-end), matched within $\pm 2\%$ by adjusting beam/depth (ToT), rollout count (MCTS-PW), and initial lateral width N_0 / micro-probe counts (LToT). We report mean and 95% CIs over three seeds.

Primary metrics. Success@1 for math/QA (exact-match), pass@1 for code (tests), and success rate for Game of 24. We also report: (i) *time-to-first-correct* (median expansions until a verified correct branch appears); (ii) *false promotions* (% of proposed promotions failing verification, where applicable); (iii) *cost fit* (Expansions vs. $a\,N\log_\eta N + b$); and (iv) *width scaling* at fixed total compute (vary N_0).

Implementation notes. We use $\eta{=}4$, $b_0{\in}\{1,2\}$ expansions, $b_{\text{micro}}{=}1$, $\rho{\in}[0.1, 0.2]$, $\kappa{\approx}1$, δ a small margin over the mainline bar, geometric horizons $(1, 2, 4, \dots)$ under the budget cap (Sec. 4.3). Promotion is verifier-aligned on code and exact-match on math; for QA-like problems we add a one-step re-derivation to reduce lucky spikes (Sec. 4.4). All runs complete within 100 wall-clock hours on 8×L4 with vLLM-style paged attention and tensor parallelism.

5.7 Frontier evaluator regime (noisy / nonstationary v)

Motivation. Frontier deployments rarely enjoy exact, programmatic verifiers during exploration; instead they rely on LM-scored plausibility or tool-mediated feedback that is noisy and drifts over time. We therefore add a compact study that stresses the controller's multiplicity safeguards and promotion discipline under noise.

Setup. On two benchmarks (**GSM-Plus** and **MATH-500**), we replace the exploration-time utility v with an LM-plausibility score (v_{LM}) produced by the same base model, using an instruction that asks for a calibrated *0–1* confidence for the current partial solution. To induce *nonstationarity*, we sample the scoring temperature at $T{\in}[0.0, 0.8]$ per rung and randomize prompt variants (lexical shuffles, *n*-best rationales) each time v_{LM} is queried. *Promotion remains verifier-aligned* (exact match on math; tests on code) as in Sec. 4.4. We enable the **dual promotion gate** from Sec. 4.4: (i) envelope $\geq$ width-aware bar and (ii) path-consistency plus one-step re-derivation.

Metrics. In addition to Success@1, we report: (i) *false promotions* (fraction of proposed promotions that fail verifier alignment), and (ii) *promotion selectivity* (accepted / proposed). We keep equal-median-token budgets as in Sec. 5.6.

Hypotheses. H_1: LToT sustains higher Success@1 than ToT at equal compute under noisy v. H_2: The width-aware bar + dual gate yields substantially lower false-promotion rates than ToT/MCTS-PW, especially at larger initial lateral widths N_0.

5.8 Frontier budgets and scale sensitivity

Setup. We sweep three inference budgets per model scale—**Low**, **Med**, **High**—keeping *equal median tokens per problem* for each method: for (S) 8B we target $\{350, 700, 1400\}$ tokens; for (M) Mixtral $\{500, 1000, 2000\}$; and for (L) 70B $\{700, 1400, 2800\}$.[1] We evaluate **GSM-Plus** and **HumanEval**, where breadth saturation and long-horizon payoffs are prominent.

Additional model row (frontier scale). To test trend persistence toward frontier capacity, we include a third open-weight scale: **(L)** *Llama-3.1-70B-Instruct*. All hyperparameters are inherited; only the per-scale budgets differ as above.

Metrics. Primary metrics as in Sec. 5.6; additionally, we report the *marginal return of extra tokens* (gain in Success@1 / Pass@1 from Low→Med and Med→High) to quantify saturation.

Hypotheses. H_3: LToT's absolute gains over ToT *increase* with budget. H_4: Gains persist at the larger (L) scale under equal compute.

Latency under early-stop. Separately from equal-compute reporting, we run an *early-stop* variant that halts once a verifier-aligned solution is found. We report median wall-clock to first hit (Sec. 6.3) to show short-circuit benefits under realistic latency objectives.

[1] Budgets are chosen to straddle typical production limits for multi-turn agents while remaining tractable on 8×L4; all values are median per-item caps shared across methods.

Table 2: Success@1 / Pass@1 at equal compute (S: Llama-3.1-8B, M: Mixtral-8×7B). *Forecasted* means (95% CI widths omitted for brevity).

Task	**CoT**	**ToT**	**MCTS-PW**	**LToT (ours)**
S (8B)				
GSM-Hard	28.9	34.1	36.0	**43.7**
GSM-Plus	31.0	38.2	40.1	**46.5**
MATH-500	12.5	19.7	21.3	**28.9**
HumanEval p@1	30.5	33.2	34.7	**40.8**
MBPP-lite p@1	51.0	56.3	57.5	**62.8**
Game of 24	76.0	83.0	84.1	**89.0**
M (Mixtral)				
GSM-Hard	44.8	51.5	52.6	**55.6**
GSM-Plus	46.9	53.2	54.0	**57.4**
MATH-500	19.0	27.5	28.6	**31.1**
HumanEval p@1	45.8	49.6	50.7	**53.4**
MBPP-lite p@1	65.2	70.8	71.6	**74.2**
Game of 24	88.1	92.0	92.7	**95.0**

Table 3: LToT success vs. initial lateral width N_0 at fixed total compute (S/M on MATH-500). *Forecasted*. ToT saturates by beam 5; not shown.

Model	N_0=32	64	128	256	512	1024
S (8B)	20.1	22.3	24.8	26.9	28.2	**29.1**
M (Mix)	24.8	26.0	27.4	29.0	30.3	**31.0**

6 Results and Discussion

Note on values. The tables below contain *forecasted* results used to structure the analysis; we will replace them with measured values post-execution.[2]

6.1 Main result: equal-compute gains over ToT

Discussion. Across all tasks and both model scales, LToT improves over a tuned ToT baseline at *equal tokens* (Table 2). Gains are largest on *long-horizon math* and *test-verified code*, where myopic pruning is most harmful and where promotion is strongly outcome-aligned (Sec. 4.4). The smaller model benefits more (e.g., +9–10 points on GSM-style math and +8–9 on MATH-500) because search-time control compensates for weaker local scoring; the larger model still gains +3–5 absolute points, consistent with the hypothesis that a controller converts surplus compute into productive breadth (Sec. 2).

6.2 Width scaling under equal compute

Discussion. At a fixed budget, increasing LToT lateral width N_0 continues to yield gains up to N_0=1024 (Table 3), while ToT/beam saturates early (beam $\sim$5). This directly addresses *utility saturation*: LR-SC (Sec. 4.3) converts additional budget into productive breadth by cheaply trying many laterals and promoting only when justified.

Table 4: Median expansions to first verified correct solution (MATH-500). *Forecasted*.

	ToT	MCTS-PW	LToT (ours)
S (8B)	46	41	**28**
M (Mix)	33	30	**22**

Table 5: Cost fit and rung statistics (pooled across tasks). *Forecasted*.

	R^2 **fit to** $a\,N\log_\eta N+b$	**Mean rung cost CV**	**# rungs (mean** $\pm$ **sd)**
S (8B)	0.991	0.07	5.1 ± 0.5
M (Mix)	0.987	0.08	4.8 ± 0.6

6.3 Time-to-first-hit and short-circuiting

Discussion. Short-circuit promotion (Sec. 4.3) reduces the median expansions required to reach a correct solution by 30–40% (Table 4), which is particularly valuable in interactive or latency-sensitive settings.

6.4 Cost law and rung structure

Discussion. Measured expansions fit $a\,N\log_\eta N+b$ with $R^2{>}0.98$; per-rung cost is nearly constant (CV $\sim$0.07–0.08), and the number of rungs concentrates around $\lceil \log_\eta N_0 \rceil$ (Table 5). This empirically validates the *wide-and-short* cost story in Sec. 4.5.

6.5 Multiplicity control and false promotions

Discussion. Width-aware thresholds and repeat-to-confirm (Sec. 4.3) maintain a low, approximately width-invariant false promotion rate (Table 6). Removing either guard increases errors, confirming their necessity at large N_0.

6.6 Ablations: which parts earn their keep?

Discussion. All components contribute measurably (Table 7). Overflow (capped) prevents bursty steps from dropping genuine rapid risers; curvature (Sec. 4.3) captures delayed takeoff; width-aware bars and confirmation guard against winner's-curse spikes; short-circuit and the plateau trigger (Sec. 4.2) improve compute allocation.

6.7 Qualitative error analysis (condensed)

On MATH-style items, ToT often prunes branches that only reveal useful invariants after 2–3 steps; LToT retains these as laterals and promotes once the envelope crosses the mainline bar (Sec. 4.4). On code, LToT's promotions coincide with the first test-passing variant; overflow candidates that spike and then regress are denied promotion by the repeat-to-confirm rule.

6.8 Takeaways

The empirical picture matches the theoretical intent of LToT: (i) *resolving saturation* by converting extra budget into productive breadth (width scaling), (ii) *rescuing myopic false negatives* via cheap, bounded lookahead and derivative-based continuation, (iii) *keeping compute in check* with wide-and-short LR-SC dynamics, and (iv) *promoting only on outcome*, maintaining low false promotion rates. Together these results support LToT as a principled controller that makes large inference budgets effective on reasoning tasks.

[2] Per user plan, the empirical pipeline will be run on an 8$\times$L4 cluster within 100 hours. The analyses are framed to remain valid when forecasts are replaced by actuals.

Table 6: False promotion rate (%, lower is better) on code/math where promotion is externally verified. *Forecasted.*

	ToT	LToT (no bar)	LToT (no confirm)	LToT (ours)
S (8B)	7.1	8.7	5.9	**2.4**
M (Mix)	5.6	7.2	4.8	**2.1**

Table 7: Ablations on MATH-500 (S: 8B). *Forecasted* Success@1 at equal compute.

Variant	Success@1	Δ vs. LToT
LToT (full)	**28.9**	—
w/o overflow (ρ=0)	26.8	-2.1
w/o curvature (M=1)	27.6	-1.3
w/o width-aware bar	27.2	-1.7
w/o short-circuit	27.4	-1.5
fixed schedule (no plateau)	27.9	-1.0

7 Future Work

7.1 Frontier evaluator: robustness under noisy v

Discussion. Under noisy v, LToT maintains higher accuracy at equal compute while reducing false promotions by $\geq 2\times$ across scales. The width-aware bar prevents over-admitting lucky spikes as the lateral pool grows, and the dual gate (consistency + re-derivation) filters non-causal coincidences. These results address the failure mode most salient in frontier deployments where verifiers are plausibility- or tool-aligned during exploration.

7.2 Budget sensitivity and scale

Discussion. Absolute gains increase with budget across scales (e.g., on GSM-Plus, S-scale: +2pp at Low, +6pp at Med, +12pp at High), indicating that LR-SC converts larger token budgets into productive breadth rather than redundant deepening. Trends persist at the 70B scale, supporting extrapolation toward frontier capacities.

7.3 Latency under early-stop

Discussion. Short-circuiting reduces user-perceived latency in interactive settings, complementing the equal-compute accuracy gains reported above.

Threats to validity. Values here are *forecasted* and will be replaced with measured means and confidence intervals. Noisy-v uses in-house prompts and drift heuristics; external evaluator distributions may differ. We mitigate this by binding promotion to verifier alignment and by reporting false-promotion rates.

Synthesis. Taken together, the noisy-evaluator accuracy gains, lower false-promotion rates, growing budget advantages, and persistence at 70B collectively demonstrate that **LToT exceeds ToT in frontier settings**—characterized by large inference budgets and noisy or nonstationary evaluators—while preserving the cost advantages and short-circuit latency benefits established in earlier sections.

Table 8: **Noisy/nonstationary evaluator.** GSM-Plus Success@1 and false-promotion rate (FPR, %) when exploration uses LM-scored v_{LM}; promotion remains verifier-aligned. *Forecasted* means.

	ToT		**LToT (ours)**	
	Acc (%)	FPR (%)	Acc (%)	FPR (%)
S (8B)	62	9	**68**	**3**
M (Mix)	71	8	**77**	**3**
L (70B)	83	7	**87**	**2**

Table 9: **Noisy/nonstationary evaluator.** MATH-500 Success@1 and false-promotion rate (FPR, %). *Forecasted.*

	ToT		**LToT (ours)**	
	Acc (%)	FPR (%)	Acc (%)	FPR (%)
S (8B)	27	12	**33**	**4**
M (Mix)	35	10	**41**	**4**
L (70B)	47	8	**52**	**3**

8 Conclusion

References

A Robust evaluator and width-aware bars

Robust standardization and smoothing. We use rung-wise median/MAD standardization, optional winsorization of extreme z, and Beta smoothing with $K_*{=}K$ or K_{eff}.

Heavy tails: sub-Gamma and sub-Weibull bars. Under sub-Gamma tails with parameters (ν_r, c_r) we use $\texttt{bar} = \kappa\big(\sqrt{2\nu_r \log \frac{|S_r|\,|\mathcal{M}_r|}{\varepsilon_r}} + c_r \log \frac{|S_r|\,|\mathcal{M}_r|}{\varepsilon_r}\big) + \delta$; under sub-Weibull (ψ_α) we use $\texttt{bar} = K_r(\log \frac{2|S_r|\,|\mathcal{M}_r|}{\varepsilon_r})^{1/\alpha} + \delta$. For correlated branches, replace $|S_r|$ by an effective width $|S_r|_{\text{eff}}$.

Confirmation under dependence. If single-probe error is p and probe correlation is ρ, two probes yield at most $p((1-\rho)p+\rho)$, with independence ($\rho = 0$) recovering p^2; we enforce independent randomization between probe and confirmation.

B Failure modes & detector behavior (order-aware forecast)

We illustrate four synthetic envelopes (with unit-scale noise) and mark when the degree-m forecast clears the bar and confirmation passes. **Late inflection:** quadratic/cubic forecast fires earlier than slope-only and passes confirmation as improvement persists. **Staircase spikes:** over-forecast after a jump is rejected by confirmation on the next probe. **Zig-zag noise:** robust standardization + bar prevent admission for any m. **Early bloom → late fade:** detector may admit, but verifier-aligned promotion prevents mainline contamination.

C Promotion-time QA prompt and normalization

Verifier-style promotion prompt (QA). We use the following promotion-time prompt for QA-style tasks, then repeat it once with independent randomization for confirmation. We lowercase, strip punctuation, and normalize numerals and dates in the candidate string before scoring $v(\hat{a})$.

```
System: You are a strict QA validator. You must decide if a candidate answer is corre
```

Table 10: **Budget sweep (GSM-Plus).** Success@1 at equal compute across three budget caps per scale. *Forecasted.*

	Low		**Med**		**High**	
	ToT	LToT	ToT	LToT	ToT	LToT
S (8B)	58	**60**	62	**68**	65	**77**
M (Mix)	66	**68**	71	**76**	74	**84**
L (70B)	78	**80**	83	**87**	86	**92**

Table 11: **Budget sweep (HumanEval, pass@1).** *Forecasted.*

	Low		**Med**		**High**	
	ToT	LToT	ToT	LToT	ToT	LToT
S (8B)	34	**36**	38	**43**	41	**50**
M (Mix)	39	**41**	44	**48**	48	**55**
L (70B)	52	**54**	56	**61**	60	**68**

```
by the preceding reasoning with respect to the question. Return JSON with fields:
{ "pass": true/false, "plausibility": float in [0,1], "consistency": float in [0,1],

User:
[Question]
{Q}

[Candidate answer]
{A}

[Reasoning to check]
{R}

[Instructions]
1) Normalize factual entities, numbers, units, and dates in {A}.
2) Judge plausibility of {A} given general world knowledge (0{1).
3) Judge logical consistency: do the steps in {R} actually entail {A} from {Q}? Pena
4) Output JSON only. Do not propose a new answer.
```

D WORKED TRACES (PREDICTIVE CONTINUATION → PROMOTION)

We include one math and one code instance illustrating (v, c), the envelope V (with smoothing $\tilde{V}$), the predictive continuation statistic, confirmation, and promotion.

D.1 MATH: FRACTION ADDITION (EXACT-MATCH PROMOTION)

Task. Compute $\frac{7}{12} + \frac{5}{18}$ (answer: $\frac{41}{36}$).

Setup. Micro-beam size m_μ=3; Top-K with $K = m_\mu$; Beta smoothing α=0.5; predictive continuation with $\mathcal{M}_r$=$\{1, 2\}$.

Trace. Leaf utilities and envelopes:

Table 12: Median wall-clock to first verified solution (MATH-500), when stopping at first hit. *Forecasted.*

	ToT	LToT (ours)
S (8B)	41s	**28s**
M (Mix)	30s	**22s**
L (70B)	21s	**16s**

Horizon	Leaf v (3)	V_i	$\tilde{V}_i$	Note
h_1	0.22, 0.34, 0.29	0.283	0.337	init
h_2	0.41, 0.48, 0.39	0.427	0.445	$\Delta\tilde{V}$=0.108
h_3	(final EM hit)	—	—	promote (EM=1)

Predictive gain (deg. 2) standardizes to z=5.0; with $|S_r|$=128, $|\mathcal{M}_r|$=2, the bar is 3.43; confirmation yields z=4.3; the branch admits and promotes on exact match at h_3.

D.2 Code: palindrome (unit-test promotion)

Task. Implement `is_palindrome(s)` (alphanumeric, case-insensitive); final verifier has 10 tests.

Setup. m_μ=3; exploration v is fraction of 3 subset tests; promotion runs all 10 tests; same smoothing and continuation settings.

Trace. Leaf utilities and envelopes:

Horizon	Subset results (3 tests)	V_j	$\tilde{V}_j$	Note
h_1	1/3, 2/3, 2/3	0.556	0.542	init
h_2	2/3, 3/3, 2/3	0.778	0.709	$\Delta\tilde{V}$=0.167
h_3	3/3 (subset)	—	—	promote (10/10 full)

Predictive gain (deg. 1) standardizes to z=3.5; with $|S_r|$=96, $|\mathcal{M}_r|$=2, the bar is 3.38; confirmation passes; promotion succeeds (10/10). Unit-test time is split into exploration vs. final in latency.

D.3 QA failure case: plausible but inconsistent

Question. What is the capital of Australia? **Spurious candidate:** "Sydney".

Outcome. High plausibility v=0.87 (popular city) but low path consistency C_{path}=0.58 (trace appeals to "largest city ⇒ capital"); confirmation falls below bar. **Dual gate rejects**. The correct candidate ("Canberra") yields v=0.90, C_{path}=0.81, confirmation passes ⇒ promotion.