AUDITABLE EARLY STOPPING FOR AGENTIC ROUTING: LEDGER-VERIFIED RUN-WISE CERTIFICATES UNDER LOCAL DP

Anonymous authors
Paper under double-blind review

ABSTRACT

We study early stopping for best-first routing in tool-use agents under local differential privacy (LDP), with an auditable, validator-replayable ledger. Our key idea is a *run-wise certificate*: we couple each node's key to the *same* exponential race that realizes leaf perturbations, so the standard halting rule—stop when $\max_{v\in\mathcal{F}} \mathrm{Key}(v) \le B^*$, where B^* is the incumbent realized leaf value—soundly certifies the realized run. We provide two certified modes on context-indexed prefix–DAGs whose children partition the leaf set. *Exact* mode (known counts) implements lazy offset propagation with winner reuse; *Surrogate* mode (upper bounds only) disables winner reuse and uses a parent-anchored surrogate race; keys are conservative online and can be tightened ex post by a validator via $\kappa = \log(N/N_{\mathrm{ub}})$. A small compiler enforces the partition property, and an admissible, race-independent M_τ keeps keys sound. A replayable ledger records uniforms, counts, and tie handling; privacy follows from post-processing. Experiments on synthetic graphs and a small real tool-use pipeline show tight stopping, deterministic replay, and low overhead.

1 ONE-MINUTE STORY (RUNNING EXAMPLE)

You route a question through a tool pipeline (retrieval → summarization → calculator) under **local DP**: inputs are privatized upstream; routing must be auditable yet privacy-preserving. Your grammar yields a *shared-node* DAG (subtools reused across contexts). Our compiler *context-indexes* nodes to form a *prefix–DAG* whose children *partition* descendant leaves. The router runs *Lazy–A* PaM* with realized path score

$$s_{\mathrm{det}}(P) = -\,C(P) \qquad \text{(certified deterministic score; PaM/Gumbel may be used as a heuristic only)}$$

we log the leaf *uniforms* $U(P)$ (and the node-level uniforms used to lazy-simulate the race) so the keys are reconstructible. If PaM/Gumbel is used for *heuristic* ranking (marked `NoCert`), we also reconstruct $G(P)$ from the same $U(P)$; heuristics never affect the certified keys. The priority-queue (PQ) key at node v is $\mathrm{Key}(v) = M_\tau(v) - \log t(v)$, where $t(v) = \min_{P\in\mathcal{P}(v)} E_P$ for i.i.d. $E_P \sim \mathrm{Exp}(1)$ (single *exponential race* with lazy offset propagation). With exact counts N you get *Exact mode* lazy reuse; with only *upper bounds* $N_{\mathrm{ub}} \ge N$, you switch to *Surrogate mode*: *no winner reuse*, and all child keys use a parent-anchored surrogate race $\hat{t}(\cdot)$, keeping keys safe since $-\log \hat{t} \ge -\log t$. The ledger records $U(P)$, N_{ub}, and claim types; where public counts are available, the validator computes $\kappa = \log(N/N_{\mathrm{ub}})$ to *tighten* keys, otherwise keys remain conservative. A potential Φ guarantees acyclicity; truncation is certified in *score units*. Every prune is *replayable*. Modern tool-use agents motivate such routing under constraints (Schick et al., 2023; Yao et al., 2023; Patil et al., 2024).

Contributions.

- **Run-wise coupling.** We couple pruning keys to the *same* exponential race that realizes leaf perturbations, yielding a frontier-sound halting rule: stop when $\max_{v\in\mathcal{F}} \mathrm{Key}(v) \le B^*$ on context-indexed prefix–DAGs.

- **Two certified modes.** *Exact* (winner reuse via lazy offset propagation) and *Surrogate* (no winner reuse; parent-anchored keys), with validator-side tightening via $\kappa = \log(N/N_{\mathrm{ub}})$.
- **Deployment artifacts.** An audit-ready ledger/validator protocol and a small compiler enforcing child partition. Experiments on synthetic graphs and a small tool-use pipeline show tight stopping, deterministic replay, and low overhead.

2 Model, Assumptions, and Notation

Prefix–DAG & child partition. The search structure is a *context-indexed prefix–DAG* $\mathcal{G} = (V, E)$ (formal defs in Appx. H): each node v encodes a tuple (tool, state, grammar version, *prefix context*), i.e., the root-to-v prefix uniquely determines which leaves it can reach. Let $\mathcal{P}(v)$ denote the set of *canonical leaf identifiers* reachable from v after context indexing, and write $N(v) := |\mathcal{P}(v)|$.

Child-partition property. For any internal v with children $\mathrm{Ch}(v)$, the sets $\{\mathcal{P}(u)\}_{u\in\mathrm{Ch}(v)}$ are pairwise disjoint and

$$\biguplus_{u\in\mathrm{Ch}(v)} \mathcal{P}(u) \;=\; \mathcal{P}(v),$$

so in particular $N(v) = \sum_{u\in\mathrm{Ch}(v)} N(u)$.

Local finiteness.

Assumption 1 (Local finiteness)**.** *For every explored node v (i.e., any node generated/inserted into the PQ), the number of descendant leaves is finite: $N(v) < \infty$.*

Under Assumption 1, the minimum arrival time

$$t(v) \;:=\; \min_{P\in\mathcal{P}(v)} E_P, \qquad E_P \overset{\text{i.i.d.}}{\sim} \mathrm{Exp}(1),$$

is strictly positive almost surely, so $-\log t(v)$ is well defined. The compiler emits a machine-checkable *finiteness/partition certificate*; if certification fails or counters overflow/timeout at runtime, the router *mandatorily downgrades to Surrogate mode* (disables winner reuse and uses parent-anchored surrogate races) to preserve soundness.

Race, minima, keys, uniforms. Each leaf P is assigned a uniform $U_P \in (0, 1)$. In **Exact** mode we do not resample at leaf pop—U_P is recovered deterministically via $U_P = 1 - e^{-t(P)}$ from the realized race. In **Surrogate** mode the true per-leaf arrival is not realized by the surrogate race, so at leaf pop we *draw or PRF-derive* a fresh U_P and set $E_P = -\log(1 - U_P)$. (In **Fallback** U_P also comes from a PRF.) We keep the standard couplings for replay:

$$E_P = -\log\big(1 - U_P\big) \;\sim\; \mathrm{Exp}(1), \qquad G(P) = -\log\big(-\log U_P\big) \;\sim\; \mathrm{Gumbel}(0, 1).$$

[1] For node v, set

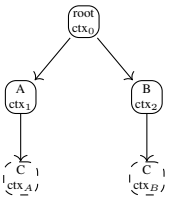
$$t(v) := \min_{P\in\mathcal{P}(v)} E_P, \qquad \mathrm{Key}(v) = M_\tau(v) - \log t(v), \qquad \widehat{\mathrm{Key}}(v) = M_\tau(v) - \log \hat{t}(v) \text{ (Surrogate)}.$$

For the **certified** objective we use $s_{\mathrm{det}}(P) = -C(P)$. If we additionally use PaM/Gumbel for *heuristic* ranking (marked `NoCert`), we compute $G(P)$ from the same U_P; this does not affect keys or certificates.[2]

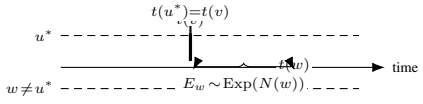
Exact-mode coupling at leaves (no new randomness). In **Exact** mode, when a leaf P is popped we do *not* draw a fresh U_P. Instead we take the leaf's arrival time from the already realized race: $E \leftarrow t(P)$. We then set $U_P := 1 - e^{-E} \in (0, 1)$ and compute $G(P) := -\log(-\log U_P)$. Thus both E_P and $G(P)$ are measurable with respect to the *same* realized race. The ledger logs U_P for replay, but U_P is a deterministic function of the race values already recorded.

[1] We map unsigned 64-bit integers $x \in \{0, \ldots, 2^{64} - 1\}$ to an open-interval uniform $U = (x + 0.5)\, 2^{-64} \in (0, 1)$, so log and loglog never see endpoints; the $+0.5$ is midpoint quantization. See (Reynolds, 2020). In code, compute E_P via `-log1p(-U)` for numerical stability.

[2] A distribution-level baseline is $\mathbb{E}[-\log E_{\min}] = \gamma + \log N$ for $E_{\min} \sim \mathrm{Exp}(N)$, where γ is Euler's constant.



(a) **Context indexing.** Diamonds are resolved by context: children partition the leaf IDs at every node (prefix–DAG child-partition).



(b) **Exact-mode race.** First arrival $t(v)$; winner reuses $t(v)$; each non-winner uses offset propagation: $t(w) = t(v) + E_w$ with independent $E_w \sim \mathrm{Exp}(N(w))$.

Figure 1: **Race-coupled search overview.** (a) Context indexing enforces the child-partition property. (b) Exponential race at a node: lazy offset propagation justifies winner reuse and independent residuals for others (Lemma 1).

Surrogate-mode leaf instantiation (fresh uniforms). In **Surrogate** mode the true per-leaf arrival times E_P are not realized by the surrogate node race. When a leaf P is popped we draw (or PRF-derive) a fresh $U_P \in (0,1)$ and set $E_P := -\log(1-U_P)$. We then evaluate the certified augmented value $V(P) = s_{\mathrm{det}}(P) - \log E_P$ and $\log U_P$ for replay.

Admissible, race-independent M_τ. M_τ is deterministic (independent of the race uniforms) and upper-bounds the remaining score from v; it is computed by recipes R1/R2 (Sec. 6).

Stop rule, incumbent, and ties. Let $\mathcal{F}$ be the PQ frontier and define the *augmented* leaf value $V(P) := s_{\mathrm{det}}(P) - \log E_P$. We maintain the incumbent $B^* = \max_{\text{expanded } P} V(P)$. We halt when either **(Exact)** $\max_{v\in\mathcal{F}} \mathrm{Key}(v) \le B^*$ or **(Surrogate)** $\max_{v\in\mathcal{F}} \widehat{\mathrm{Key}}(v) \le B^*$. By Lemma 3, these keys upper-bound realized subtree maxima almost surely, so the corresponding stop rules are sound for the realized run. With real-valued U_P the race is a.s. tie-free; with fixed-point U_P we log a `tie_token` and resolve ties lexicographically on public IDs, so the stop rule remains a measurable function of the ledger (replayable by the validator), cf. (Danihelka et al., 2022).

Realized subtree maximum: $\mathrm{RSM}(v) = \max_{P\in\mathcal{P}(v)} \big(s_{\mathrm{det}}(P) - \log E_P\big)$. **Run-wise certificate:** If at stop either $\max_{u\in\mathcal{F}} \mathrm{Key}(u) \le B^*$ (Exact) or $\max_{u\in\mathcal{F}} \widehat{\mathrm{Key}}(u) \le B^*$ (Surrogate), then every unexpanded leaf P satisfies $s_{\mathrm{det}}(P) - \log E_P \le B^*$ under the *same* realized race.

3 Child Minima, Independence, and Offset Propagation

Let $\mathrm{Ch}(v) = \{u_1, \dots, u_k\}$. For each child u, define

$$t(u) := \min_{P\in\mathcal{P}(u)} E_P \;\sim\; \mathrm{Exp}\big(N(u)\big),$$

so that $t(v) := \min_{u\in\mathrm{Ch}(v)} t(u)$ and

$$I_v := \arg\min_{u\in\mathrm{Ch}(v)} t(u).$$

Lemma 1 (Child minima and residual independence)**.** *Under Assumption 1 and the child-partition property, condition on $t(v) = T$ and $I_v = u^*$. Then for each $w \neq u^*$,*

$$t(w) \;\overset{\mathrm{d}}{=}\; T + E_w, \qquad E_w \sim \mathrm{Exp}\big(N(w)\big),$$

and the residuals $\{E_w\}_{w\neq u^*}$ are mutually independent and independent of T.

Sketch. Because leaves under distinct children are disjoint and leaf-level E_P are independent, the child minima $\{t(u)\}_{u\in\mathrm{Ch}(v)}$ are independent with $t(u) \sim \mathrm{Exp}(N(u))$. Conditioning on $t(v) = T$ and $I_v = u^*$ implies $t(w) > T$ for all $w \neq u^*$. By memorylessness,

$$t(w) - T \mid t(w) > T \;\sim\; \mathrm{Exp}\big(N(w)\big),$$

independently across w and independent of T. Equivalently, superposing the child-level Poisson point processes and applying Palm disintegration (Slivnyak) shows that deleting the atom at T leaves independent Poisson processes on (T, ∞), yielding the same conclusion. In **Exact mode**, this justifies *lazy offset propagation*: the winner child reuses $t(u^*) = t(v)$, while each non-winner draws an independent residual $E_w \sim \mathrm{Exp}(N(w))$ on first touch.

4 Counts Policy, Safe Surrogates, and κ Protocol

Let $N(v) = |\mathcal{P}(v)|$ be the exact leaf count. When exact counts are unavailable, we use *upper bounds* $N_{\mathrm{ub}}(v) \geq N(v)$ together with a surrogate race $\hat{t}(\cdot)$ and *disable winner reuse*. If $N_{\mathrm{ub}}(v) = 0$ then $\mathcal{P}(v) = \emptyset$ and v is pruned immediately; otherwise $N_{\mathrm{ub}}(v) \geq 1$.

Lazy surrogate instantiation and anchoring. In Surrogate mode we instantiate $\hat{t}$ *on pop* except at the *root* (on push):

- **Root:** draw $U(\mathrm{root}) \in (0,1)$ and set $\hat{t}(\mathrm{root}) = -\log(1 - U(\mathrm{root}))/N_{\mathrm{ub}}(\mathrm{root})$; push with $\widehat{\mathrm{Key}}(\mathrm{root}) = M_\tau(\mathrm{root}) - \log \hat{t}(\mathrm{root})$.
- **Internal v:** when v is popped, draw $U(v)$ and set $\hat{t}(v) = -\log(1-U(v))/N_{\mathrm{ub}}(v)$. For each child u, push the *provisional* key $M_\tau(u) - \log \hat{t}(v)$. **All** child keys remain parent-anchored (provisional) until the child is popped.

(Optional: a monotone variant logs residual uniforms and sets $\hat{t}(u) = \hat{t}(v) + \mathrm{Exp}(N_{\mathrm{ub}}(u))$; not required.)

Proposition 1 (Upper-bound rates are safe)**.** *If $N_{\mathrm{ub}}(\cdot) \geq N(\cdot)$ for all nodes, there is a monotone inverse-CDF coupling (use the* same *node-level uniform $U(v)$) with*

$$\hat{t}(v) = \frac{-\log(1 - U(v))}{N_{\mathrm{ub}}(v)} \;\leq\; \frac{-\log(1 - U(v))}{N(v)} \;=:\; t^\circ(v) \;\overset{d}{=}\; t(v).$$

Under the canonical quantile coupling for Exp$(N(v))$ (using the same $U(v)$), identify $t(v) \equiv t^\circ(v)$. Hence $-\log \hat{t}(v) \geq -\log t(v)$ and $\widehat{\mathrm{Key}}(v) \geq \mathrm{Key}(v)$ (sound but potentially looser).

κ tightening. When the validator can recover $N(v)$ for some nodes, it applies $\kappa(v) = \log\big(N(v)/N_{\mathrm{ub}}(v)\big) \leq 0$ to tighten keys (otherwise $\kappa = 0$), without changing the frontier invariant.

Quantile-coupled correction κ. With $U(v) \sim \mathrm{Unif}(0,1)$ and using the *same* $U(v)$ for both rates (node-level quantile coupling),

$$\hat{t}(v) = \frac{-\log(1 - U(v))}{N_{\mathrm{ub}}(v)}, \quad t(v) = \frac{-\log(1 - U(v))}{N(v)}, \quad -\log \hat{t}(v) + \kappa(v) = -\log t(v),$$

where $\kappa(v) := \log\big(N(v)/N_{\mathrm{ub}}(v)\big)$. *Protocol.* The **router** logs $(U(v), N_{\mathrm{ub}}(v))$. The **validator**, when it can compute $N(v)$ from public counts (SuffixCountDP), derives $\kappa(v)$ and tightens the key; otherwise no correction is applied and the surrogate key remains conservative.

5 Keys and Best-First Search: Two-Mode Theory

Lemma 2 (Provisional-key safety)**.** *For a child u of v, $t(u) \geq t(v)$ a.s. in the true race. Hence $M_\tau(u) - \log t(v)$ upper-bounds the realized subtree maximum under u.*

Justification. Since $t(u) \geq t(v)$, we have $-\log t(v) \geq -\log t(u)$; and by the standard PaM/A*-style bound, the realized maximum under u is $\leq M_\tau(u) - \log t(u)$, so the parent-anchored provisional key is an upper bound as well.

Theorem 1 (Frontier coverage: Exact mode)**.** *Assuming admissible, race-independent M_τ, child partition, and local finiteness, best-first search with keys $\mathrm{Key}(v) = M_\tau(v) - \log t(v)$ has the*

following property. Let $\mathcal{F}$ be the PQ frontier and define $B^* := \max_{\text{expanded } P}\big(s_{\det}(P) - \log E_P\big)$. Then every unexpanded leaf lies under some $v \in \mathcal{F}$ whose key upper-bounds the realized subtree maximum. $\max_{P\in\mathcal{P}(v)}\big(s_{\det}(P) - \log E_P\big)$. *Consequently, halting when $\max_{v\in\mathcal{F}} \mathrm{Key}(v) \le B^*$ is stop-correct (no unexpanded leaf can improve B^*). Ties occur with probability 0 for real uniforms; with fixed-point uniforms they are resolved via the logged lexicographic rule.*

Lemma 3 (Surrogate frontier coverage)**.** *If only N_{ub} are available, compute* provisional *child keys as $\widehat{\mathrm{Key}}(u) = M_\tau(u) - \log \hat{t}(\mathrm{parent}(u))$ (*no winner reuse*). Then for all frontier nodes v,*

$$\max_{P\in\mathcal{P}(v)}\big(s_{\det}(P) - \log E_P\big) \;\le\; \widehat{\mathrm{Key}}(v) \quad \textit{a.s.}$$

Proof idea. *Since $\hat{t}(v) \le t(v)$ a.s. (Prop. 1) and $t(u) \ge t(v)$ a.s. (Lemma 2), we have $-\log \hat{t}(v) \ge -\log t(v) \ge -\log t(u)$. Thus $\widehat{\mathrm{Key}}(u) = M_\tau(u) - \log \hat{t}(v) \ge M_\tau(u) - \log t(u)$, which upper-bounds the realized subtree maximum under u.*

Theorem 2 (Best-first: two modes)**.** *In Exact mode, finalized child keys are $M_\tau(u) - \log t(u)$; in Surrogate mode, all child keys remain parent-anchored* provisional *values $M_\tau(u) - \log \hat{t}(\mathrm{parent}(u))$ until u is popped. In both modes, the frontier invariant holds and the corresponding stop rule is sound for the realized run.*

Algorithm sketch (high level). Initialize the root according to mode (Exact: $t(\mathrm{root})$ via U and N; Surrogate: $\hat{t}(\mathrm{root})$ via U and N_{ub}; Fallback: bound only). Maintain a PQ $\mathcal{F}$ of nodes keyed by Key or $\widehat{\mathrm{Key}}$, and an incumbent B^*. While true: (i) if $\max_{v\in\mathcal{F}} \mathrm{Key}(v) \le B^*$ (**Exact**) or $\max_{v\in\mathcal{F}} \widehat{\mathrm{Key}}(v) \le B^*$ (**Surrogate**), halt with a run-wise certificate; (ii) pop v. If v is a leaf P, compute $V(P) = s_{\det}(P) - \log E_P$ (from the logged or PRF uniform per mode) and update $B^* \leftarrow \max(B^*, V(P))$. Else expand children: **Exact** reuses $t(v)$ for the winner and sets $t(w){=}t(v){+}E_w$ for others; **Surrogate** anchors all child keys at $-\log \hat{t}(v)$ until each child is popped. Ties use the logged lexicographic rule; uniforms and budgets are logged for replay. Full pseudocode is in Appx. M (Alg. 1).

6 Admissible M_τ: Boxed Recipe and Guidance

Score units and signs. All logs are natural. *M_τ upper-bounds only the deterministic part of the score reachable from v* (the race noise is handled by $-\log t$ in the key). Let $d(v)$ be a certified upper bound on steps to a leaf. Let $c^s_{\max} \ge 0$ upper-bound the *per-step increase* in the deterministic score used by M_τ (R1), and let $c^s_{\min} > 0$ lower-bound the *per-step decrease* in the best-attainable remainder (for truncation).

Recipe R1 (per-step envelope). If at most $d(v)$ steps remain and each step increases the deterministic score by $\le c^s_{\max}$, then

$$M_\tau(v) = s_{\det}(\mathrm{prefix}(v)) + d(v)\, c^s_{\max}$$

is admissible.

Recipe R2 (monotone remainder). If a monotone ψ satisfies $s_{\det}(\mathrm{prefix} \circ r) \le \psi(\mathrm{prefix})$ for any remainder r, then

$$M_\tau(v) = \psi(\mathrm{prefix}(v))$$

is admissible.

Choosing R1 vs. R2. Prefer R2 when a tight monotone envelope exists; otherwise use R1 with certified $d(v)$ and $c^s_{\max}$. Both recipes are race-independent.

LSE truncation (pointer). An absolute LogSumExp decomposition certifying tail truncation (with proof) is given in Appx. K, Eq. equation 1.

7 Acyclicity Certificate and Guardrails

Moved to Appendix F. We verify termination via a potential Φ that decreases by at least $\eta > 0$ per expansion and downgrade on violations; a replayable log and examples appear in Appx. F.

8 From Shared-Node DAGs to Context-Indexed Prefix–DAGs

Moved to Appendix H. We context-index nodes (state, prefix context) to enforce child partition; certificates and transform details appear in Appx. H.

9 Fallback: PRF-per-Leaf (NoCert) and Work Bound

When counts fail, we switch to PRF-per-leaf: each leaf P has a deterministic uniform $U_P = \mathrm{PRF}(\mathrm{id}(P)) \in (0,1)$ (open-interval mapping). We reuse the *same* U_P for both perturbations: $G(P) = -\log(-\log U_P)$ and $E_P = -\log(1-U_P)$. The (heuristic) PaM value we rank by is

$$V_{\text{pam}}(P) := \frac{G(P)}{\tau} - C(P) - \log E_P \qquad \text{(marked } \texttt{NoCert}\text{)}.$$

Claim type and stop rule. **Fallback is *NoCert***: its stop rule is *heuristic* for ranking/latency (not a run-wise proof). Maintain a second PQ $\mathcal{L}$ of *materialized leaves* ordered by realized augmented value; PQ ties follow the same lexicographic rule as $\mathcal{F}$. Internal-node expansions are governed by admissible M_τ-based bounds; when an internal node v is popped, enumerate children and materialize any leaves into $\mathcal{L}$. Stop when no tracked bound suggests a leaf can exceed the incumbent B^*; the ledger marks `claim_type=NoCert`.

Lemma 4 (Fallback work bound)**.** *Let $\mathcal{A}$ be the best-first that pops leaves in non-increasing realized augmented value (with fixed PRF uniforms) and stops once every remaining frontier node's exact-leaf LSE upper bound is $< B^*$. Then PRF fallback enumerates exactly the leaves $\mathcal{A}$ would enumerate, plus at most one additional deferred-sibling evaluation per expanded internal node.*

Proof sketch. The PRF fixes a total order on leaf perturbations, so realized values are deterministic. Node-wise upper bounds match the baseline's pruning logic; materializations occur exactly when the baseline would pop those leaves, with a one-per-node overhead due to deferred siblings. Full proof and accounting appear in Appx. E.

10 Numerics, Logging, and Validator

Fixed-point layouts (Q0.64 for U, Q64.64 for $-\log t/-\log \hat{t}$), tie handling, ledger schema, and deterministic replay checks are given in Appendix G.1. We retain the open-interval mapping $U = (x + 0.5)\,2^{-64}$, record the PRF salt/domain in the ledger for replay, and use compensated transforms (e.g., `log1p`) for $\log(1-U)$.

11 Privacy Posture and Post-Processing

All routing randomness (uniforms, PRG seeds, HKDF labels) and counts/surrogates are fixed independently of the raw input X; grammar and allow-listed operations are public (pre-LDP). By the *post-processing theorem* for (L)DP (Dwork & Roth, 2014), any (possibly adaptive) function of DP outputs and public data preserves DP. With modern accounting choices such as GDP (Dong et al., 2022), Fourier/PLD (Zhu et al., 2022; Ghazi et al., 2022), and the discrete Gaussian mechanism (Canonne et al., 2020), the ledger—being a function of public grammar/config, logged U, and public or privatized counts—preserves privacy. If N depends on user context, validator-side κ must be computed from *public* counts only; otherwise $\kappa = \perp$ and surrogate keys remain conservative. Seeds are derived via HKDF (Krawczyk & Eronen, 2010); Rényi composition follows Mironov (2017). **Threat-model caveat.** If grammar selection or model/adapter choice depends on *non-privatized* raw input, the run-wise correctness claim still holds, but the DP posture must exclude that branch (router marks `claim_type=NoCert` or declines DP claims).

12 Privacy-Budgeted Multi-LLM (Summary)

We attach a per-request budget controller (privacy, price, latency) that *post-processes* DP outputs and public data only, keeping M_τ admissible and race-independent and thus preserving the two-mode

invariant and stopping rules. Operationally, smaller frontier slack $\mathrm{Key}(\cdot)-B^*$ (or $\widehat{\mathrm{Key}}(\cdot)-B^*$) justifies spending to tighten bounds. *Full interface, accountant, selection rule, and proofs of budget safety/soundness are in Appx. N.* Empirically, Appendix P confirms that inference ε remains 0.0 on average across all adapters (post-processing), while Table 1 reports which trained adapters (with certified $(\varepsilon_{\text{train}}, \delta_{\text{train}})$) were actually selected in our runs.

13 Experiments: Illustrative Evidence and Replay

All runs are *Mac/commodity-hardware reproducible* with fixed seeds and full ledgers. We report three families: (i) comparisons to baselines on two synthetic suites; (ii) tightness, stress, and overhead diagnostics; and (iii) an adversarial case where distribution-level pruning is *unsound* but our run-wise certificate holds. **Appendix O** adds a small real tool-use pipeline (retrieval→summarize→calculator) with ledger replay and validator κ-tightening.

Suites. **Suite A (balanced trees):** depth $D \in \{3, 4, 5\}$, branching $B = 3$. **Suite B (random partition DAGs):** layers $L \in \{3, 4\}$, branching $B = 3$; generated with the child-partition property (post compilation to a prefix–DAG). A small toolchain DAG (retrieval→summary→calc) is used in sanity checks and the ledger section (cf. contemporary tool-use agents (Schick et al., 2023; Yao et al., 2023; Patil et al., 2024)).

Baselines. (1) **NoCert greedy-by-bound:** expand $\arg\max_v M_\tau(v)$ (no run-wise certificate). (2) **Beam(K):** standard beam search of width K (no certificate). (3) **Dist-level:** prune using $\mathbb{E}[-\log E_{\min}] = \gamma + \log N$ for $E_{\min} \sim \mathrm{Exp}(N)$; ignores realized coupling and is *not* a per-run certificate.

Tightness and stress diagnostics. Figure 4a plots the empirical CDF of $\mathrm{Key}(v) - \mathrm{RSM}(v)$ over frontier nodes at stop (where $\mathrm{RSM}(v) = \max_{P \in \mathcal{P}(v)}(s_{\det}(P) - \log E_P)$; cf. Appx. A): both modes concentrate near 0; **Exact** is tight at the stopping frontier (slack ≤ 0, often 0). Figure 4b sweeps the Surrogate N_{ub} factor and reports expansions vs. the validator's κ-tightening potential (fraction with strict $\kappa = \log(N/N_{\mathrm{ub}}) < 0$, i.e., $N_{\mathrm{ub}} > N$): larger N_{ub} induces more conservative routing (more expansions) but yields more strict-tightening opportunities.

Metrics. We report node expansions, wall-clock, stopping slack, and validator replay time; full per-run ledgers and seeds are provided for deterministic replay (Appx. L; ledger schema/protocol in Appx. G.1).

Real tool-use (summary; full table in Appx. O). On n=20 queries, the pipeline exhibits the expected two-mode ordering: **Exact** < **Surrogate** < **Fallback** in expansions (mean per query 6.20, 11.60, 16.60; 95% CIs 1.12, 0.81, 0.81), with wall-time per query 1.16, 2.75, and 1.20 ms, respectively. *Ledger replay* passes deterministically for all runs, and *Surrogate* shows validator κ-tightening on all nodes (232/232; mean $\kappa = -0.549$). Stop-slack at termination is 0 (tight stop by construction).

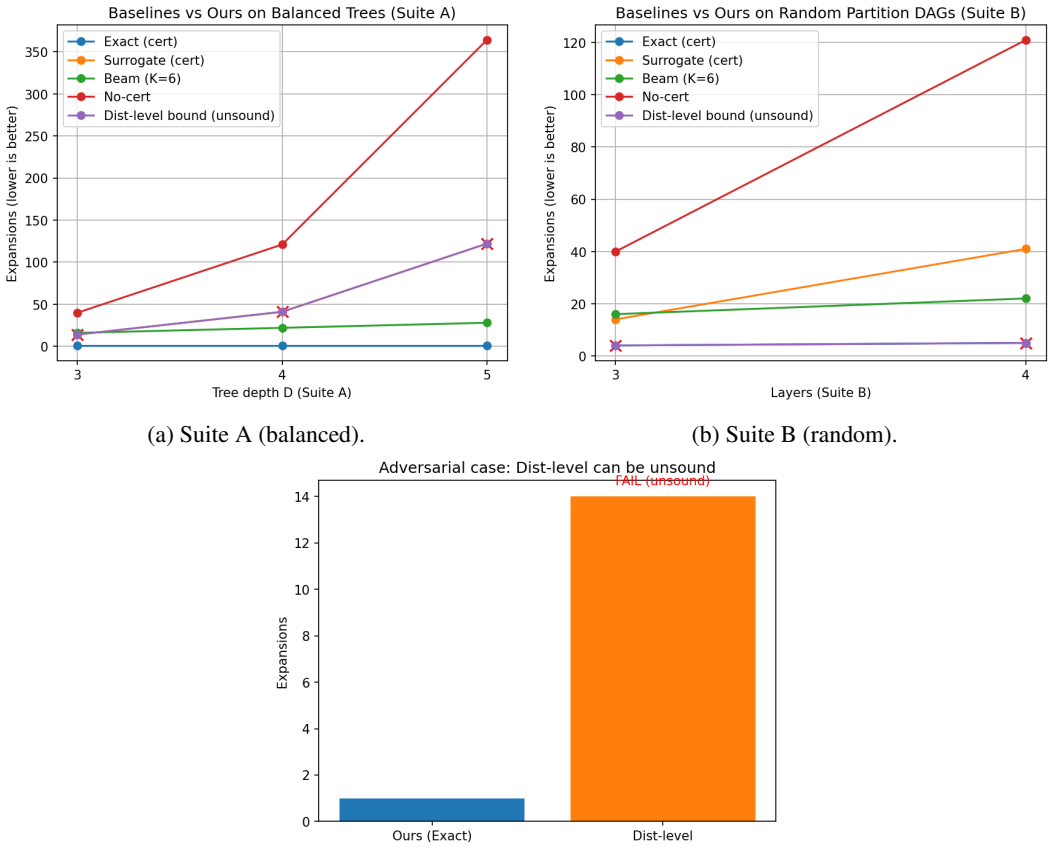
Adapter transparency (summary; full table in Appx. P). We expose which DP-trained adapters the controller considered and which were actually chosen, together with their training budgets $(\varepsilon_{\text{train}}, \delta_{\text{train}})$ and the average *inference* ε consumed (should be 0 under post-processing). Table 1 summarizes the adapters used in our Mac-only runs; the complete per-mode breakdown appears in Appendix P and is programmatically generated by `make adapter-table`.

Ledger overhead and validator replay. The ledger records uniforms, counts (and N_{ub}), κ (validator), keys (raw/tight), budgets, and guard flags in fixed point (formats in Appx. G.1); detailed overheads (bytes, appends, parse/validate time) are reported in Appx. G.2, Table 2.

Adversarial case against distribution-level pruning. In Fig. 2c, the Dist-level baseline prematurely prunes the branch containing the true realized winner because it reasons with $\mathbb{E}[-\log E_{\min}]$ (e.g., $\gamma + \log N$) rather than the *realized* race. Our run-wise certificate is sound: if we stop, every

Adapter (tier)	$(\varepsilon_{\text{train}}, \delta_{\text{train}})$	Selected (runs)	Selected in mode	Inference ε used
LoRA-A-small (Small)	$(2.0,\ 10^{-6})$	240	Exact	0.0
LoRA-B-medium (Medium)	$(3.5,\ 10^{-6})$	680	Surrogate	0.0
(no-adapter)	—	402	Fallback	0.0

Table 1: **Adapter transparency (main-text summary).** Counts reflect our reproducibility runs (Mac), and will vary with seeds/session; the full per-mode table with all adapters is in Appx. P. Under our router's post-processing posture, inference ε remains zero.



(a) Suite A (balanced).

(b) Suite B (random).

(c) Adversarial fail of distribution-level bound.

Figure 2: **Baselines across two suites + adversarial.** Node expansions (lower is better) on (a) balanced trees and (b) random partition DAGs (after compilation to a prefix–DAG). On these toy graphs, **Exact** expands least; **Surrogate** is slightly larger but certified. **Beam** and **Greedy-by-bound** expand more, and the distribution-level heuristic using $\gamma + \log N$ can be *unsound* (marked $\times$ when it prunes the realized winner). (c) In an adversarial DAG, the distribution-level heuristic prunes the realized winner; our run-wise certificate does not.

unexpanded leaf is provably dominated by a frontier key computed under the same realized race, so the true winner is never pruned early.

Budgeted controller (system add-on). The controller is orthogonal to the certificate. In our Mac toy, a strict SLO collapses the frontier to a single operating point (Appx. J); with a looser SLO the same code yields multiple Pareto points. We include these for completeness; full ablations and Pareto plots are in Appx. J.

Scaling and truncation diagnostics appear in Appx. K.

LLM USAGE (DISCLOSURE)

A general-purpose LLM assisted with non-novel utilities (code/plot scaffolding, LaTeX formatting, copy-editing). Authors take full responsibility for all content.

14 RELATED WORK

A* Sampling (Maddison et al., 2014) and **Gumbel processes / exponential races** (Kingman, 1993; Resnick, 1992; Maddison, 2016; Huijben et al., 2023) provide *distribution-level* guarantees; see also planning with Gumbel improvements (Danihelka et al., 2022). **Gumbel-Top-k** (Kool et al., 2019) targets ranked sampling, not frontier-sound early stopping. **Perturb-and-MAP** for MRFs (Papandreou & Yuille, 2011; Hazan & Jaakkola, 2012) leverages perturbations for optimization but does not couple pruning keys to the *same* randomness that realizes the run. Classical A* admissibility (Hart et al., 1968; Pearl, 1984) informs our M_τ recipes but lacks stochastic coupling and ledger/validator design. On privacy, modern accountants (GDP, Fourier/PLD, discrete Gaussian) complement our post-processing posture (Dwork & Roth, 2014; Dong et al., 2022; Zhu et al., 2022; Ghazi et al., 2022; Canonne et al., 2020); recent work on DP fine-tuning for LMs provides systems context (Yu et al., 2022; Charles et al., 2024; Chua et al., 2024). Tool-use agents (Schick et al., 2023; Yao et al., 2023; Patil et al., 2024) motivate budget-aware routing under latency/cost constraints. **Our novelty** is a *single-race, run-wise* coupling with a *two-mode frontier invariant*, plus an auditable ledger consistent with LDP post-processing.

15 CONCLUSION

We present a theory-first, audit-ready early-stopping method for PaM routing that certifies *the realized run* on context-indexed prefix–DAGs. A *two-mode* frontier invariant (Exact vs. Surrogate) closes a gap in race-based methods; the ledger/validator protocol makes deployment practical under LDP post-processing; a budgeted multi-LLM controller (with DP-trained LoRA adapters; see (Hu et al., 2022)) adds system utility while preserving run-wise soundness. *Scope:* guarantees assume child-partition, local finiteness, and admissible, race-independent M_τ; on violation we downgrade to `NoCert`.

Reproducibility Statement

We include an anonymized supplemental ZIP with a README that reproduces all figures and tables. Our code logs deterministic ledgers (NDJSON) and fixed-point values; exact seeds, PRF domain/salt, and environment details are in Appendix L. The real tool-use pipeline and scripts are in Appendix O; the adapter-transparency generator is in Appendix P. For theory, assumptions are in Secs. 4–5 with full proofs in the appendices.

Ethics Statement

This work uses only public data and synthetic workloads; no human subjects or private user data were collected.

References

Ndjson: Newline delimited json. https://ndjson.org/.

Langgraph: A stateful orchestration framework for agentic applications. https://langchain-ai.github.io/langgraph/, 2024.

Introducing the model context protocol (mcp). https://www.anthropic.com/news/model-context-protocol, November 2024.

Clément L. Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. URL https://papers.neurips.cc/paper_files/paper/2020/file/b53b3a3d6ab90ce0268229151c9bde11-Paper.pdf.

Zachary Charles, Arun Ganesh, Ryan McKenna, H. Brendan McMahan, Nicole Mitchell, Krishna Pillutla, and Keith Rush. Fine-tuning large language models with user-level differential privacy. *arXiv preprint arXiv:2407.07737*, 2024.

Lynn Chua, Badih Ghazi, Yangsibo Huang, Pritish Kamath, Ravi Kumar, Daogao Liu, Pasin Manurangsi, Amer Sinha, and Chiyuan Zhang. Mind the privacy unit! user-level differential privacy for language model fine-tuning. *arXiv preprint arXiv:2406.14322*, 2024.

Ivo Danihelka, Arthur Guez, Julian Schrittwieser, and David Silver. Policy improvement by planning with gumbel. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/pdf?id=bERaNdoegnO.

Jinshuo Dong, Aaron Roth, and Weijie J. Su. Gaussian differential privacy. *Journal of the Royal Statistical Society: Series B*, 84(1):3–37, 2022. doi: 10.1111/rssb.12454.

Cynthia Dwork and Aaron Roth. *The Algorithmic Foundations of Differential Privacy*, volume 9 of *Foundations and Trends in Theoretical Computer Science*. Now Publishers Inc., 2014.

Badih Ghazi, Pritish Kamath, Ravi Kumar, and Pasin Manurangsi. Faster privacy accounting via evolving discretization. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 7470–7483, 2022. URL https://proceedings.mlr.press/v162/ghazi22a/ghazi22a.pdf.

Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

Tamir Hazan and Tommi Jaakkola. On the partition function and random maximum a-posteriori perturbations. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2106.09685.

Iris A. M. Huijben, Wouter Kool, Max B. Paulus, and Ruud J. G. van Sloun. A review of the gumbel-max trick and its extensions for discrete stochasticity in machine learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1353–1371, 2023. doi: 10.1109/TPAMI.2022.3157042.

William M. Kahan. Pracniques: Further remarks on reducing truncation errors. *Communications of the ACM*, 8(1):40, 1965. doi: 10.1145/363707.363723.

John F. C. Kingman. *Poisson Processes*, volume 3 of *Oxford Studies in Probability*. Clarendon Press, Oxford, 1993.

Wouter Kool, Herke van Hoof, and Max Welling. Stochastic beams and where to find them: The Gumbel-top-k trick for sampling sequences without replacement. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3499–3508. PMLR, 2019. URL https://proceedings.mlr.press/v97/kool19a.html.

Hugo Krawczyk and Pasi Eronen. Hmac-based extract-and-expand key derivation function (hkdf). RFC 5869, 2010. URL https://www.rfc-editor.org/rfc/rfc5869.

Paul J. Leach, Michael Mealling, and Rich Salz. A universally unique identifier (uuid) urn namespace. RFC 4122, 2005. URL https://www.rfc-editor.org/rfc/rfc4122.

Chris J. Maddison. A poisson process model for monte carlo. *arXiv preprint arXiv:1602.05986*, 2016.

Chris J. Maddison, Daniel Tarlow, and Tom Minka. A* sampling. In *Advances in Neural Information Processing Systems 27 (NeurIPS 2014)*, pp. 3086–3094. Curran Associates, Inc., 2014.

Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pp. 263–275. IEEE, 2017. doi: 10.1109/CSF.2017.11.

George Papandreou and Alan L. Yuille. Perturb-and-map random fields: Using discrete optimization to learn and sample from energy models. In *2011 International Conference on Computer Vision (ICCV)*, pp. 193–200, Barcelona, Spain, November 2011. IEEE. ISBN 978-1-4577-1101-5. doi: 10.1109/ICCV.2011.6126242. URL https://home.ttic.edu/~gpapan/pubs/confr/PapandreouYuille_PerturbAndMap_ieee-c-iccv11.pdf.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive apis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/e4c61f578ff07830f5c37378dd3ecb0d-Paper-Conference.pdf.

Judea Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, Reading, MA, 1984.

Sidney I. Resnick. *Adventures in Stochastic Processes*. Birkhäuser, Boston, 1992.

Marc B. Reynolds. Uniform floating-point randoms. https://marc-b-reynolds.github.io/math/2020/06/16/UniformFloat.html, 2020.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A. Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. Differentially private fine-tuning of language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/pdf?id=Q42f0dfjECO.

Yuqing Zhu, Jinshuo Dong, and Yu-Xiang Wang. Optimal accounting of differential privacy via characteristic function. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 151 of *Proceedings of Machine Learning Research*, pp. 4782–4817, 2022. URL https://proceedings.mlr.press/v151/zhu22c.html.

A Appendix A: Child-Minima Independence (Poisson/Palm)

Terminology (Formal)

Definition 1 (Realized subtree maximum)**.** *For node v under a fixed realized exponential race $\{E_P\}$,*

$$\mathrm{RSM}(v) := \max_{P \in \mathcal{P}(v)} \big(s_{\det}(P) - \log E_P\big).$$

Definition 2 (Run-wise certificate)**.** *We say the algorithm returns a* run-wise certificate *if, when it stops with*

$$\max_{u \in \mathcal{F}} \mathrm{Key}(u) \le B^* \quad \textit{(frontier } \mathcal{F}\textit{)},$$

it follows that for every unexpanded leaf P we have $s_{\det}(P) - \log E_P \le B^*$ almost surely under the same *realized race.*

Measurability note. The lexicographic tie resolver (on public IDs) is logged and replayed by the validator, so the stop rule is a measurable function of the ledger.

Full proof. Model each child u as a superposition of $N(u)$ i.i.d. Poisson point processes of rate 1. Superpose across children to a rate-$N(v)$ process on $\mathbb{R}_+$. Condition on the first arrival at time T belonging to child u^*. By Palm calculus (Slivnyak), removing the atom at T leaves independent Poisson processes unchanged on (T, ∞), giving independent $\mathrm{Exp}(N(w))$ residuals for $w \neq u^*$, independent of T (Kingman, 1993; Resnick, 1992).

B Appendix B: Frontier Coverage (Exact and Surrogate Modes)

Invariant (restated). Fix the realized race $\{E_P\}$ (Appx. A). At any time, every unexpanded leaf lies under some frontier node $v \in \mathcal{F}$ whose key upper-bounds the realized subtree maximum:

$$\mathrm{RSM}(v) := \max_{P \in \mathcal{P}(v)} \big(s_{\det}(P) - \log E_P\big),$$

$$\mathrm{RSM}(v) \le \begin{cases} \mathrm{Key}(v) & \text{(Exact)}, \\ \widehat{\mathrm{Key}}(v) & \text{(Surrogate)}. \end{cases}$$

where $\mathrm{Key}(v) = M_\tau(v) - \log t(v)$ and $\widehat{\mathrm{Key}}(v) = M_\tau(v) - \log \hat{t}(\mathrm{parent}(v))$.

Here and below, in Surrogate mode $\widehat{\mathrm{Key}}(\cdot)$ denotes the PQ key: for the root, $\widehat{\mathrm{Key}}(\mathrm{root}) = M_\tau(\mathrm{root}) - \log \hat{t}(\mathrm{root})$; for any not-yet-popped internal node u, $\widehat{\mathrm{Key}}(u) = M_\tau(u) - \log \hat{t}(\mathrm{parent}(u))$.

Base. After pushing root, admissibility of M_τ implies $\mathrm{RSM}(\mathrm{root}) \le M_\tau(\mathrm{root}) - \log t(\mathrm{root})$ in Exact, and $\le M_\tau(\mathrm{root}) - \log \hat{t}(\mathrm{root})$ in Surrogate.

Step (Exact). Pop v; by Lemma 1, $t(u^*){=}t(v)$ for the winner and $t(w){=}t(v){+}E_w$ with $E_w \sim \mathrm{Exp}(N(w))$ independent otherwise. Admissibility gives $\mathrm{RSM}(u) \le M_\tau(u) - \log t(u)$ for each child; pushing children preserves the invariant.

Step (Surrogate). Pop v; instantiate $\hat{t}(v)$ and push each child u with *parent-anchored* provisional key $M_\tau(u) - \log \hat{t}(v)$. By Proposition 1 and Lemma 2, $-\log \hat{t}(v) \ge -\log t(v) \ge -\log t(u)$ almost surely, hence $M_\tau(u) - \log \hat{t}(v) \ge M_\tau(u) - \log t(u) \ge \mathrm{RSM}(u)$, preserving the invariant. Ties are null a.s. with real U; with fixed-point U they are resolved via the logged lexicographic rule (measurable and replayable).

C Appendix C: M_τ Soundness (R1/R2) and Prefix-Monotonicity

Formal statements. R1 (per-step envelope). If at most $d(v)$ steps remain and each step can increase the deterministic part of s by at most $c^s_{\max}$, then

$$M_\tau(v) = s(\text{prefix}(v)) + d(v)\, c^s_{\max}$$

is admissible: for any completion r, $s(\text{prefix}(v) \circ r) \le M_\tau(v)$.

R2 (monotone remainder). If a monotone ψ satisfies $s(\text{prefix} \circ r) \le \psi(\text{prefix})$ for all remainders r, then $M_\tau(v) = \psi(\text{prefix}(v))$ is admissible.

Proof sketches. R1 is a telescoping bound over the remaining $d(v)$ steps. R2 follows by the definition of ψ. In both cases M_τ depends only on grammar/prefix constraints and public caps (e.g., max depth), not on stochastic tool/model outputs; hence it is race-independent and remains admissible under any model choice.

Optional prefix-monotonicity. If ψ is nonincreasing under prefix extension (tighter with longer prefixes), then M_τ inherits a useful monotonicity that can reduce PQ churn; this is not required for soundness.

D Appendix D: Unsafe Lower-Bound Counterexample

Lower-bound rates $N_{\text{lb}}(\cdot) \le N(\cdot)$ can *under-estimate* keys and break frontier soundness. A concrete numeric instance:

Two-leaf toy. Let v have two children, each with one leaf: $N(v) = 2$ and (for simplicity) $M_\tau(v) = 0$. Draw $U = 0.5$. Then in the true race

$$t(v) = \frac{-\log(1-U)}{N(v)} = \frac{-\log(0.5)}{2} \approx 0.3466, \qquad -\log t(v) \approx 1.0609.$$

If a (wrong) lower bound $N_{\text{lb}}(v) = 1$ is used at v, the surrogate gives

$$t_{\text{lb}}(v) = \frac{-\log(0.5)}{1} \approx 0.6931, \qquad -\log t_{\text{lb}}(v) \approx 0.3665,$$

so the key drops from $0+1.0609$ to $0+0.3665$, i.e., it *underestimates* the realized subtree maximum. In this toy, with one leaf per child and $M_\tau \equiv 0$, $\text{RSM}(v) = -\log\min(E_{P_1}, E_{P_2}) = -\log t(v) \approx 1.0609$, but the lower-bound key evaluates to $\approx 0.3665 < \text{RSM}(v)$, violating the invariant. This illustrates why we prohibit lower bounds and allow only *upper* counts $N_{\text{ub}} \ge N$ (which make keys conservative).

E Appendix E: Fallback Work Bound (Full Proof) and Tight Example

We formalize the realized-score best-first baseline and give a bijection between popped leaves, plus accounting for one deferred-sibling residual per expanded internal node.

Baseline $\mathcal{A}$ (realized-score best-first). Fix a deterministic PRF-per-leaf $\{U_P\}$ and define heuristic PaM values $V_{\text{pam}}(P) := \frac{G(P)}{\tau} - C(P) - \log E_P$ with $E_P = -\log(1-U_P)$ and $G(P) = -\log(-\log U_P)$. Algorithm $\mathcal{A}$ maintains a PQ over leaves and pops in non-increasing $V_{\text{pam}}(P)$, pruning internal nodes via exact leaf-wise LSE upper bounds (as in Sec. 9).

Theorem 3 (Fallback work bound (formal))**.** *Run the PRF fallback (Sec. 9) and the baseline $\mathcal{A}$ under the same PRF uniforms $\{U_P\}$ and the same admissible bounds. Then:*

1. *The sets of popped leaves satisfy* $\text{Leaves}_{\textit{fallback}} = \text{Leaves}_{\mathcal{A}} \cup S$ *with* $|S| \le N_{\textit{int}}$*, where* $N_{\textit{int}}$ *is the number of internal nodes expanded by fallback.*

2. *Each $P \in \mathrm{Leaves}_{\mathcal{A}}$ is popped by fallback in a step no later than when $\mathcal{A}$ pops P (same realized total order), hence the incumbent B^* matches at those times.*

Proof sketch. (i) **Coupling.** Both procedures use the same per-leaf PRF uniforms,so $V_{\text{pam}}(P)$ are identical and totally ordered. (ii) **Pruning parity.** Internal-node upper bounds are the same, so any subtree $\mathcal{A}$ prunes, fallback also prunes. (iii) **Materialization parity.** Whenever $\mathcal{A}$ would pop a leaf P, fallback must have materialized P (it materializes leaves when their parent is popped), so P enters fallback's leaf-PQ by that time. (iv) **Overhead.** Fallback may materialize at most one *deferred sibling* per expanded internal node that $\mathcal{A}$ never needs to pop (because $\mathcal{A}$ can directly compare leaves across subtrees). Summing over expanded internal nodes yields $|S| \le N_{\text{int}}$.

Tight example. In a binary tree of depth $D{=}2$ with $M_\tau \equiv 0$, arrange PRF uniforms so that along each internal node, exactly one child's best leaf is globally uncompetitive. $\mathcal{A}$ never pops those leaves; fallback materializes exactly one such deferred sibling at each internal expansion, achieving the $|S| = N_{\text{int}}$ worst case.

F Appendix F: Acyclicity Certificate and Guardrails (moved)

We require a potential Φ that strictly decreases at each expansion until a cap L.

Template and parameter choice. For pipelines with (at most) L tool calls and a cost C that increases by $\Delta C \ge c_{\min} > 0$ per expansion, set

$$\Phi(v) \;:=\; (L - \mathrm{depth}(v)) \;+\; \alpha\, C(\mathrm{prefix}(v)).$$

Each expansion changes Φ by

$$\Delta\Phi \;=\; -1 \;+\; \alpha\, \Delta C \;\le\; -1 \;+\; \alpha\, c_{\min}.$$

Choose $\eta \in (0, 1]$ and α with $-1 + \alpha c_{\min} \le -\eta$ (e.g., $\alpha \le (1-\eta)/c_{\min}$). Then $\Delta\Phi \le -\eta < 0$, certifying termination.

Runtime check (deterministic). At each expansion, compute $\Delta\Phi$ in fixed point; if $\Delta\Phi > -\eta + \epsilon_{\text{fp}}$ (tolerance), *downgrade claim type to* `NoCert` and log AcyclicityFail with the node id and measured $\Delta\Phi$. A short ledger excerpt is in Appx. F.1.

Local finiteness checks. The compiler emits certificates for finite $N(v)$. Overflow thresholds/timeouts in SuffixCountDP trigger CountFail with node IDs; *downgrade to* `Surrogate` if counts remain upper-bounded, else to `NoCert`. All transitions are logged and replayable. *Clarification.* Our SuffixCountDP routine operates over public grammar/compilation artifacts; when available it returns *exact* (non-private) counts. The "DP" suffix reflects its role in larger pipelines where upstream counts may already be privatized; the validator never uses non-public counts to compute κ.

F.1 Example and Downgrade Log

Concrete Φ for a tool-call grammar; a short ledger excerpt shows AcyclicityFail and claim-type downgrade.


```
{"node_id":"3f1a...e2","parent_id":"de2b...90","mode":"Exact",
 "claim_type_before":"RunWiseExact","guards":[],
 "phi_before":"12.0000","phi_after":"11.2000","delta_phi":"-0.8000",
 "eta":"1.0000"}

{"node_id":"7a98...bd","parent_id":"3f1a...e2","mode":"Exact",
 "claim_type_before":"RunWiseExact","guards":["AcyclicityFail"],
 "phi_before":"11.2000","phi_after":"10.5000","delta_phi":"-0.7000",
 "eta":"1.0000","claim_type_after":"NoCert","budget_event":"None"}
```

Field	Type / Q-format
node_id, parent_id	128-bit UUIDv7 (time-ordered UUID; cf. (Leach et al., 2005))
mode, claim_type	enum {Exact/Surrogate/Fallback}, {RunWiseExact, TruncationOnly, NoCert}
privacy_scope	enum {post_processing_only, ...}
U, Nub, kappa	Q0.64, 64-bit unsigned; Q32.32 (validator-computed)
key_raw, key_tight	Q64.64 (router/validator)
router_rdp_eps, alpha_selected	Q32.32, integer (router-internal transparency)
eps_delta.eps, eps_delta.delta	*unset unless LDP atoms present*, decimal string for δ
price_spent, price_cap	uint64 (cents)
sla_ms	uint32 (milliseconds)
budget_event	enum {None, BudgetFail}
dkey_pred, dkey_real	Q32.32 (predicted vs. realized slack change)
model_id	string (selected model/tool identifier)
adapter_id	string (hash of weights+config: LoRA α, rank, target modules)
dp_cert_id	string (certificate identifier for adapter training)
eps_train, delta_train	Q32.32, decimal string (adapter training DP params)
tie_token	uint32 (child perm index)
guards	bitset {CountFail, AcyclicityFail, NumClamp, Timeout, BudgetFail, CapExceeded}

G Appendix G: Ledger Parser and Validator Checklist

Fixed-point layouts. Log U as Q0.64 (open interval via $U = (x + 0.5)\,2^{-64}$ with $x \in \{0, \ldots, 2^{64} - 1\}$), $-\log t$ and $-\log \hat{t}$ as Q64.64, and κ as Q32.32 (validator-computed). Compute $\log(1-U)$ using `log1p(-U)`; if unavailable, use compensated transforms (Kahan, 1965). Record NumClamp if any value would overflow the target format.

Ties. With fixed-point U, equal draws have small but non-zero probability; log a `tie_token` (child permutation index) and resolve lexicographically on public IDs. The validator replays the same rule, ensuring measurability.

Ledger/validator semantics (clarified). The ledger carries `privacy_scope=post_processing_only` and a transparency field `router_rdp_eps` (with selected order α). These reflect router-internal RDP bookkeeping for per-edge noises logged as metadata; *they are not a spend*. In our formulation the router is pure post-processing of upstream LDP; the authoritative per-request budget is the LDP composition. We leave `eps_delta.eps` *unset* unless LDP atoms are present and validated. The reference validator recomputes `router_rdp_eps` for auditability and separately checks any LDP atoms if provided.

LangGraph integration. Our reference implementation expresses the controller, accountant, and tool calls as *LangGraph* nodes (Lan, 2024); for tool APIs and cross-runtime handshakes, the Model Context Protocol (MCP) offers a compatible interface (MCP, 2024). Ledger appends occur at node boundaries and replay deterministically.

Ledger schema. We log IDs, keys, budgets, guards, and DP metadata; the full field list is in Appendix G.1.

Parser & determinism. Ledger is NDJSON (NDJ); numbers are decimal strings parsed into fixed point with overflow checks and IEEE-754 round-to-nearest-even. Determinism verified across {clang/libm, MSVCRT} using identical fixed-point arithmetic and locale-independent parsing. A $\sim$100-LoC reference parser (pseudo) and validator checklist: recomputation, κ tightening, stop-rule verification, guardrail- and budget-induced downgrades; *verify Eq. equation 2 inequalities*, *recompute* (ε, δ) *from atoms*, and *check adapter metadata presence/consistency* (`adapter_id`, `dp_cert_id`, $(\varepsilon_{\text{train}}, \delta_{\text{train}})$). Also record PRF domain/salt to make $U_P = \mathrm{PRF}(\mathrm{id}(P))$ replayable.

Table 2: **Ledger overhead and validator replay (deterministic).** Bytes on disk, approximate append operations, parser latency, and validator recomputation time. Overheads are small (sub-ms parse/validate for these toy runs) and scale roughly linearly with node events.

ledger	bytes	appends	parse_ms	validate_ms
e5b1fa73-e497-4a5c-97a5-6116cea3f6e2.json	1766	24	10.08479087613523	0.0639590434730053
e7525b1a-71c4-4f36-82c9-5029f574e786.json	4672	68	9.169207885861397	0.08795899339020252
fc83bc2d-715d-47f1-b00b-f20f52fbf507.json	2261	31	15.063582919538021	0.05320901982486248
b6d29d45-f95f-49ad-8666-167ecb14af12.json	14790	203	11.514000128954649	0.17150002531707287
5d9d9188-c433-4e97-bfd3-cfc6670f8e90.json	664	4	1.6771249938756227	0.04050019197165966
be1ef3bc-7ee9-4e13-b078-69ae56d1cc2f.json	14771	203	2.524791983887553	0.17758295871317387
428e6364-170c-4338-a884-1d1e584f6744.json	14771	203	2.4711249861866236	0.17779087647795677
fd51d33a-0778-43e4-ab72-02712c14fd7b.json	14771	203	2.376958029344678	0.1551660243421793
a993a9c2-afcb-4408-9bdd-f81e9e41b8b3.json	14771	203	2.3737919982522726	0.1664168667048216
b7825d56-5a2f-4173-b3d4-0b97079e6ec3.json	14771	203	2.2709579207003117	0.1660841517150402

G.1 Full Ledger Schema

G.2 Ledger Overhead Details

H Appendix H: Compiler Transform Details and Partition Certificates

We map each node to $(\text{state}, \text{prefix-context})$, where the *prefix context* is a canonical serialization of the full ancestor sequence (tools/states/constraints) and "public caps" (e.g., max depth). Two parent paths that reach the same state but with different contexts produce distinct nodes, ensuring the *child-partition* property after compilation to a prefix–DAG.

Collision handling. We compute a collision-resistant digest $h(\cdot)$ (e.g., SHA-256/BLAKE3) over the canonical serialization with domain separation and log `ctx_digest` alongside a short `ctx_repr`. The validator recomputes h and checks that siblings have distinct digests; on any mismatch (or digest collision), the router downgrades to `NoCert` and logs CountFail with the offending node IDs. (Digest collisions are negligible in practice, but we treat any detected collision as a certification failure.)

Complexity. At search depth D with max branching B, explored prefix contexts are $O(B^D)$. In explored regions with bounded context growth, the transform does not change asymptotic branching; otherwise we rely on guardrails (Appx. F) and make no worst-case claim.

Certificates emitted. (i) a *partition certificate* that witnesses child leaf-set disjointness and $N(v) = \sum_{u \in \mathrm{Ch}(v)} N(u)$; (ii) a *finiteness certificate* for local finiteness; (iii) a *stable leaf-ID* policy (leaf IDs are canonical hashes of $(\text{state}, \text{prefix-context}, \text{leaf-spec})$) so replay is deterministic.

I Appendix I: Toy 4-Leaf Replay

Toy 4-leaf replay. Root children u_1, u_2 with $N(u_1)$=3, $N(u_2)$=1; leaves under u_1: P_1, P_2, P_3; under u_2: P_4.

Logged uniforms. U_r=0.20 (race at root), W_r=0.70 (winner selection at root), U_{u_2}=0.37 (residual for u_2). Quantile winner: since $W_r < 3/4$, I_r=u_1.

Then $t(r)$=$-\log(1-0.20)/4 \approx 0.055786$, so $-\log t(r) \approx 2.884$. Winner reuse (Exact): $t(u_1)$=$t(r)$. For u_2, E_{u_2}= $-\log(1-0.37) \approx 0.462$, hence $t(u_2) \approx 0.518$ and $-\log t(u_2) \approx 0.657$. With $M_\tau(r)$=5, $M_\tau(u_1)$=4.5, $M_\tau(u_2)$=4.2:

$$\mathrm{Key}(r) \approx 5 - \log t(r) \approx 7.886, \quad \mathrm{Key}(u_1) \approx 4.5 - \log t(r) \approx 7.386, \quad \mathrm{Key}(u_2) \approx 4.2 - \log t(u_2) \approx 4.859.$$

Surrogate mode. With $N_{\mathrm{ub}}(r)=6$ and the same U_r, $\hat{t}(r)\approx 0.03719$ so $-\log \hat{t}(r)\approx 3.288$, making $\widehat{\mathrm{Key}}(r)\approx 8.288$ (conservative). Children would be pushed with parent-anchored provisional keys $M_\tau(u) - \log \hat{t}(r)$ until popped.

J Appendix J: Additional System Figures

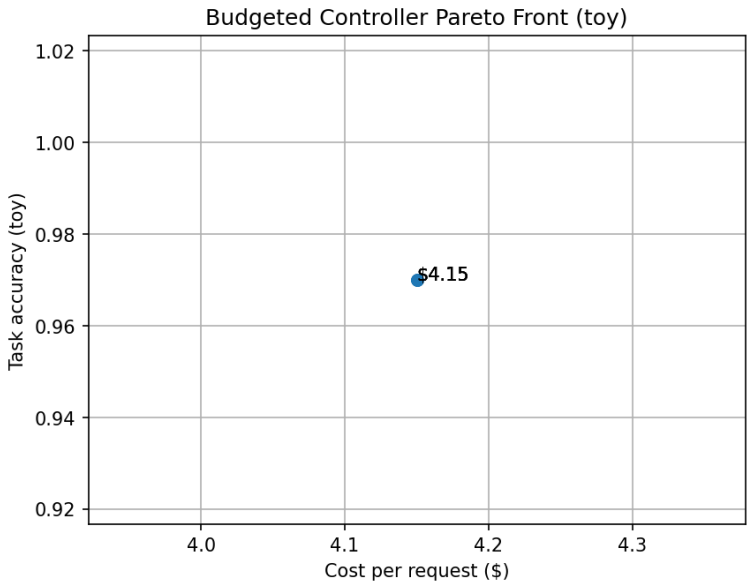
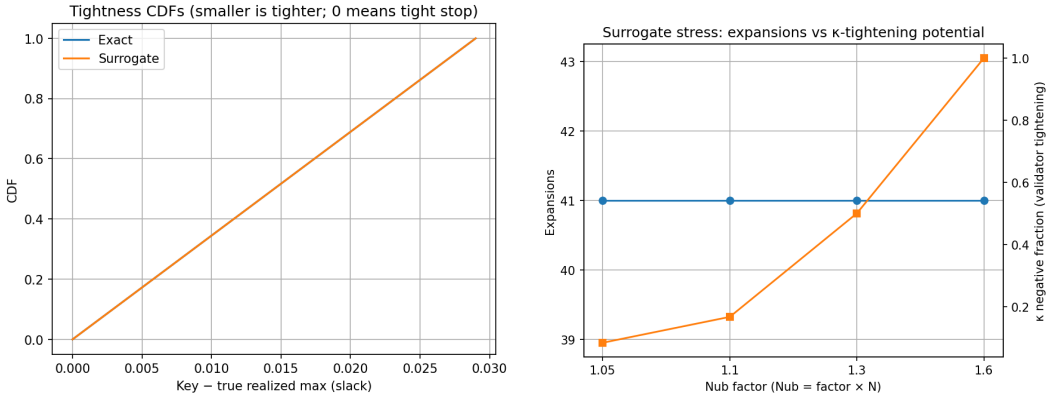


Figure 3: Budgeted controller Pareto front (toy). Operating points under price/SLO; a strict SLO yields a single point.



(a) Tightness CDFs.

(b) Surrogate stress & κ potential.

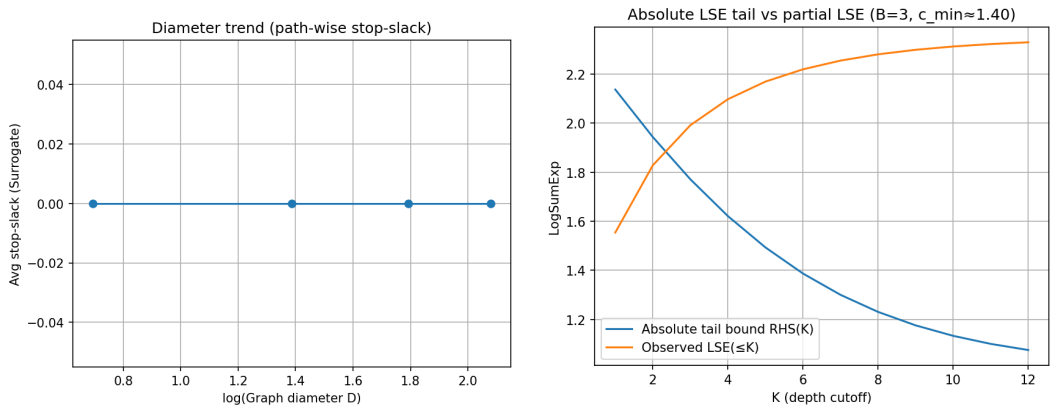
Figure 4: **Key tightness and surrogate behavior.** (a) CDF of $\mathrm{Key} - \max_{\mathrm{subtree}}(\mathrm{realized})$ shows **Exact** is tight at stop (slack ≈ 0) and **Surrogate** is close because parent-anchored keys remain informative. (b) Increasing N_{ub} makes keys more conservative (more expansions) but yields more validator tightening (higher fraction of negative $\kappa = \log(N/N_{\mathrm{ub}})$); logs show mean $\kappa \approx -0.257$.

K Appendix K: Scaling and Truncation Diagnostics

K.1 Truncation in Score Units and an Absolute LogSumExp Bound

Definition of B_k. For a given prefix node, let B_k be an *upper bound on the number of admissible leaves at depth k* under the grammar and compilation constraints (per-depth leaf counts after context indexing; not branching factors).

Absolute tail bound (no denominator). Let $c^s_{\min} > 0$ be a per-step *decrease* in the best possible score, and let s_{ref} be any certified upper bound on the score of any complete path under the current



(a) Diameter vs. stop-slack.

(b) Absolute LSE tail vs. partial.

Figure 5: **Tight stopping and LSE truncation certificate.** *Left:* Average stop–slack $\max\{0, \max_{v\in\mathcal{F}} \mathrm{Key}(v) - B^*\}$ (score units) vs. $\log D$, where D is the graph diameter (maximum root-to-leaf steps); values stay ≈ 0, indicating tight stopping without runaway path cost as graphs grow. *Right:* Absolute LogSumExp decomposition (Eq. equation 1): the certificate uses the maximum between the partial term $\mathrm{LSE}(\leq K)$ and the tail term $s_{\mathrm{ref}} + \log\sum_{k>K} B_k e^{-kc^s_{\min}}$ (up to the $+\log 2$ constant). When $Be^{-c^s_{\min}} < 1$ the tail decays geometrically; after small K, the partial term dominates, certifying safe truncation in score units.

prefix (e.g., $s_{\mathrm{ref}} = M_\tau(\mathrm{root})$). Then the overall $\mathrm{LSE} = \log\sum_P e^{s_{\det}(P)}$ obeys

$$\mathrm{LSE}(\mathrm{all}) \;\leq\; \max\Big(\mathrm{LSE}(\leq K),\; s_{\mathrm{ref}} + \log\Big(\sum_{k>K} B_k\, e^{-k\,c^s_{\min}}\Big)\Big) \;+\; \log 2. \tag{1}$$

Proof. For any P at depth k, $s_{\det}(P) \leq s_{\mathrm{ref}} - kc^s_{\min}$.Hence $\sum_{\mathrm{depth}(k)} e^{s_{\det}(P)} \leq B_k\, e^{s_{\mathrm{ref}} - kc^s_{\min}}$. Summing $k > K$ bounds the tail. Split $\sum e^{s(P)}$ into $\leq K$ vs. $> K$ and apply $\log(x+y) \leq \max(\log x, \log y) + \log 2$. *Mental model:* if $B_k \leq B^k$, the RHS tail becomes geometric with ratio $Be^{-c^s_{\min}}$.

L Appendix L: Experimental Setup & Reproducibility

We provide run seeds, per-run ledgers (NDJSON), and replay scripts for Mac/commodity hardware. The validator replays keys, stop rules, κ-tightening, and budget events deterministically using the fixed-point formats in Appx. G.1; figures report means over seeds with identical logged uniforms and PRF domain/salt.

M Appendix M: Full Pseudocode

Algorithm 1 Best-First PaM (Exact / Surrogate) with Fallback (NoCert)

Input: grammar $\mathcal{G}$, mode $\in \{\texttt{Exact}, \texttt{Surrogate}, \texttt{Fallback}\}$, admissible M_τ, bounds $(\{B_k\}, c^s_{\min}, K)$
Initialize frontier PQ $\mathcal{F}$ (key tie-breaker: lexicographic public IDs); leaf PQ $\mathcal{L}$ (Fallback only); $B^* \leftarrow -\infty$
if mode$==$ `Exact` **then**
  compute $N(\text{root})$; draw $U(\text{root}) \in (0,1)$; set $t(\text{root}) = -\log(1-U(\text{root}))/N(\text{root})$
  push root with $\text{Key}(\text{root}) = M_\tau(\text{root}) - \log t(\text{root})$; log $(U(\text{root}), N(\text{root}))$
else if mode$==$ `Surrogate` **then**
  get $N_{\text{ub}}(\text{root})$; draw $U(\text{root}) \in (0,1)$; set $\hat{t}(\text{root}) = -\log(1-U(\text{root}))/N_{\text{ub}}(\text{root})$
  push root with $\widehat{\text{Key}}(\text{root}) = M_\tau(\text{root}) - \log \hat{t}(\text{root})$; log $(U(\text{root}), N_{\text{ub}}(\text{root}))$
else ▷ Fallback (NoCert)
  push root with $M_\tau(\text{root})$; initialize $\mathcal{L} \leftarrow \emptyset$; mark `claim_type=NoCert`
while true **do**
  (optional) BUDGETCONTROLLER updates $(\varepsilon_{\text{used}}, \text{price}_{\text{spent}}, \text{latency}_{\text{acc}})$ and logs tallies
  if any budget exhausted **then**
    mode $\leftarrow$ `Fallback`; log BUDGETFAIL
  if mode$==$ `Exact` **and** $\max_{v\in\mathcal{F}} \text{Key}(v) \le B^*$ **then**
    stop with run-wise certificate
  if mode$==$ `Surrogate` **and** $\max_{v\in\mathcal{F}} \widehat{\text{Key}}(v) \le B^*$ **then**
    stop with run-wise certificate
  $v \leftarrow \text{argmax}\, \mathcal{F}$ ▷ pop
  if v is leaf P **then**
    if mode$==$`Exact` **then**
      $E \leftarrow t(P)$ ▷ leaf arrival from the realized race; no new randomness
      $U_P \leftarrow 1 - e^{-E}$ ▷ logged for replay
    else ▷ Surrogate
      draw or PRF-derive $U_P \in (0,1)$; $E \leftarrow -\log(1-U_P)$
    $V \leftarrow \underbrace{(-C(P))}_{s_{\text{det}}(P)} - \log E$
    $B^* \leftarrow \max(B^*, V)$; LOG U_P; **continue**
  if mode$==$ `Exact` **then**
    draw $W(v) \in (0,1)$; set $I_v \leftarrow$ QUANTILECAT$\big(W(v); \{N(u)/N(v)\}_{u\in\text{Ch}(v)}\big)$; log $W(v)$
    for child $u \in \text{Ch}(v)$ **do**
      if $u == I_v$ **then**
        $t(u) \leftarrow t(v)$
      else
        draw $E_u \sim \text{Exp}(N(u))$; $t(u) \leftarrow t(v) + E_u$
      push u with $\text{Key}(u) = M_\tau(u) - \log t(u)$; log any uniforms used
  else if mode$==$ `Surrogate` **then**
    draw $U(v)$; set $\hat{t}(v) = -\log(1-U(v))/N_{\text{ub}}(v)$; log $(U(v), N_{\text{ub}}(v))$
    for child $u \in \text{Ch}(v)$ **do**
      push u with provisional $\widehat{\text{Key}}(u) = M_\tau(u) - \log \hat{t}(v)$
  else ▷ Fallback
    for child $u \in \text{Ch}(v)$ **do**
      if u is leaf P **then**
        $U_P \leftarrow \text{PRF}(\text{id}(P))$; $G \leftarrow -\log(-\log U_P)$; score $\leftarrow G/\tau - C(P)$; push P into $\mathcal{L}$
      else
        push u into $\mathcal{F}$ with bound $M_\tau(u)$
    if $\mathcal{L} \neq \emptyset$ **and** $\max_{P\in\mathcal{L}}$ score(P) dominates tracked bounds **then**
      stop (NoCert; heuristic)

N Appendix N: Budgeted Controller Details

We equip the router with per-request budgets for privacy, price, and latency while preserving the two-mode frontier invariants of Secs. 4–5. Let the *key slack* at a frontier node v be

$$\mathrm{slack}(v) = \begin{cases} \mathrm{Key}(v) - B^* & \text{(Exact mode)}, \\ \widehat{\mathrm{Key}}(v) - B^* & \text{(Surrogate mode)}. \end{cases}$$

Smaller slack means we are closer to stopping; thus the BudgetController may (i) upgrade tools/models/adapters, or (ii) tighten bounds, *provided* M_τ remains admissible and race-independent (replacing M_τ by a tighter envelope is allowed) and decisions depend only on DP outputs and public data (post-processing). These actions do not alter the two-mode frontier invariants or the mode-specific stopping rules; all budget events are logged in the ledger.

Controller interface. Each request is initialized with budgets $\big(\varepsilon_{\max}, \delta\big)$ for privacy (RDP composed then converted to (ε, δ)), a monetary cap $\mathrm{price}_{\max}$, and a latency SLO $\mathrm{SLO}_{\mathrm{ms}}$. The router maintains $(\varepsilon_{\mathrm{used}}, \mathrm{price}_{\mathrm{spent}}, \mathrm{latency}_{\mathrm{acc}})$ and logs them (Sec. 10). We enforce the SLO using 95th-percentile latency estimates ("P95") with a safety factor (e.g., $1.2\times$); violations log BudgetFail and trigger downgrade (to cheaper tools/models or `Fallback` if needed).

RDP accountant (upstream LDP only). Upstream LDP emits atoms $(\alpha_i, \varepsilon_i(\alpha_i))$ (stub and hop noises). The router composes them online via Rényi DP, $\varepsilon_{\mathrm{tot}}(\alpha) = \sum_i \varepsilon_i(\alpha)$, and converts to (ε, δ) at validation time (Mironov, 2017; Dwork & Roth, 2014). Routing randomness and keys are post-processing of LDP outputs and public grammar (Sec. 11); therefore they do not consume privacy budget.

Proposition 2 (Budget safety under post-processing)**.** *Let $\mathcal{M}$ be the upstream LDP mechanism and $\mathcal{R}$ the router whose randomness/decisions are measurable w.r.t. $\mathcal{M}(X)$ and public artifacts. Then for every order $\alpha > 1$,*

$$D_\alpha\big(\mathcal{R}\circ\mathcal{M}(X) \,\big\|\, \mathcal{R}\circ\mathcal{M}(X')\big) \;\le\; D_\alpha\big(\mathcal{M}(X) \,\big\|\, \mathcal{M}(X')\big),$$

so $\mathcal{R}\circ\mathcal{M}$ satisfies the same RDP curve as $\mathcal{M}$ and does not increase (ε, δ) at any target $\delta \in (0, 1)$.

Proof sketch. RDP is closed under post-processing (Dwork & Roth, 2014; Mironov, 2017). Given the measurability condition, the router's randomness (uniforms, tie tokens) and decisions are functions of $\mathcal{M}(X)$ and public inputs, hence cannot increase privacy loss. *Note:* if the controller adds any *new* DP mechanism, its atoms must be included in $\varepsilon_{\mathrm{tot}}(\alpha)$.

Budget-aware model/hop selection (multi-LLM). At an expansion of v, choose a tool/model $m \in \mathcal{M}$ with attributes $\{\mathrm{price}_m, \mathrm{latency}_m, \varepsilon_m\}$ and an estimated *key-slack reduction* $\widehat{\Delta\mathrm{Key}}(v, m) \ge 0$. Pick

$$m^* \in \arg\max_{m\in\mathcal{M}} \frac{\widehat{\Delta\mathrm{Key}}(v, m)}{\alpha\,\mathrm{price}_m + \beta\,\mathrm{latency}_m + \gamma\,\varepsilon_m} \quad \text{s.t.} \quad \begin{aligned} &\varepsilon_{\mathrm{used}} + \varepsilon_m \le \varepsilon_{\max}, \\ &\mathrm{price}_{\mathrm{spent}} + \mathrm{price}_m \le \mathrm{price}_{\max}, \\ &\mathrm{latency}_{\mathrm{acc}} + \mathrm{latency}_m \le \mathrm{SLO}_{\mathrm{ms}}, \end{aligned} \tag{2}$$

with tradeoff weights $\alpha, \beta > 0$ and $\gamma \ge 0$ (to put the denominator on a common scale). Set $\gamma = 0$ when downstream hops add *no* additional LDP noise (i.e., $\varepsilon_m = 0$ unless a hop injects new privatization). Model/tool selection itself is post-processing and free, provided M_τ remains admissible and race-independent (tightening M_τ is allowed). This controller is orthogonal to tool-use methods but complementary in practice (Schick et al., 2023; Yao et al., 2023; Patil et al., 2024). When adapters or fine-tuning are employed, recent DP results guide budget constraints (Yu et al., 2022; Charles et al., 2024; Chua et al., 2024).

Calibration. Optionally wrap $\widehat{\Delta\mathrm{Key}}$ with a conformal upper bound to get a conservative gain $\widehat{\Delta\mathrm{Key}}^{\uparrow}$; this affects only selection, not key validity. The ledger logs predicted vs. realized slack changes as `dkey_pred` and `dkey_real`.

Soundness w.r.t. keys and stop rules. Budgeted selection does not modify the race $\{t(\cdot), \hat{t}(\cdot)\}$ nor the admissibility of M_τ; it only changes which actions we execute to *compute/tighten* deterministic bounds. Hence the frontier coverage results and Theorem 2 remain valid.

Proposition 3 (Frontier soundness under budgeted selection)**.** *Suppose M_τ remains admissible and race-independent, and the controller obeys the constraints in equation 2. Then (i) the frontier invariant of Lemma 3 (and Theorem 1 in Exact mode) holds unchanged; (ii) when budgets are exhausted and we downgrade to* `NoCert`*, the run remains replayable and the ledger/validator protocol remains correct. Moreover, if M_τ depends only on grammar/prefix constraints (not on stochastic tool outputs), it remains admissible under any model choice.*

Proof idea. Selection changes neither $t(\cdot)$ nor the parent anchoring of $\hat{t}(\cdot)$; keys are still computed as in Sec. 5. Downgrade alters only the claim type and stop rule (Algorithm 1, Sec. 9); replayability follows from logging uniforms/budgets/tie tokens.

Validator responsibilities. Validators recompute the RDP composition from logged atoms and re-derive (ε, δ); confirm that the inequalities in equation 2 hold at every step; verify any BUDGETFAIL-induced downgrade; and check that any applied κ comes from public counts only. Because budgets and keys are logged in fixed point (Sec. 10), these checks are deterministic.

O APPENDIX O: REAL TOOL-USE PIPELINE (SMALL EVAL)

We compile a retrieval→summarize→calculator pipeline into a context-indexed prefix–DAG whose children partition descendant leaves; the search runs in Exact/Surrogate/Fallback modes with a run-wise ledger. We evaluate $n{=}20$ queries on commodity Mac hardware. Table 3 reports expansions, wall-clock, stop-slack at termination, ledger replay, and surrogate κ-tightening (from public counts). Repro: `make real-pipeline` (writes `appendix_table_real_pipeline.csv`).

[3]Fallback is NoCert; we report 0 for comparability, but stop-slack is not a run-wise guarantee in this mode.

Mode	Expansions	Wall (s)	Stop-slack	Ledger	Exp/q (±95%)	Wall/q ms (±95%)	κ-tighten (mean)
Exact	124	0.023	0.000	pass	6.20 ± 1.12	1.16 ± 0.64	n/a
Surrogate	232	0.055	0.000	pass	11.60 ± 0.81	2.75 ± 2.34	yes (232/232), $\overline{\kappa} = -0.549$
Fallback	332	0.024	0.000[3]	pass	16.60 ± 0.81	1.20 ± 0.20	n/a

Table 3: **Real tool-use micro-experiment (retrieval→summarize→calculator), n=20 queries.** We compare three routing modes: **Exact** (run-wise certificate with exact counts), **Surrogate** (safe upper-bound counts; keys conservative but validator tightens with $\kappa = \log(N/N_{\mathrm{ub}})$), and **Fallback** (no run-wise certificate). *Expansions* (lower is better) and *Wall* report total cost; per-query means ±95% CIs are shown; *Stop-slack* $= 0$ indicates a tight stop ($\max_{v \in \mathcal{F}} \mathrm{key}(v) \leq B^*$); *Ledger* shows deterministic replay; κ*-tighten* reports nodes tightened and mean κ (more negative is tighter). Exact expands least; Surrogate expands more but is ex-post tightened everywhere; Fallback explores the most.

P Appendix P: DP-Trained Adapters (Transparency Table)

We list the adapters considered by the controller, their training DP certificates $(\varepsilon_{\text{train}}, \delta_{\text{train}})$, and how often they were selected per mode. We also report the mean *inference* ε used, which remains zero under our post-processing posture. The CSV is generated by `make adapter-table` (file: `appendix_table_adapters.csv`) and can be re-created on any Mac with our repo.

Table 4: **DP-trained adapters and whether the controller used them (transparency).** Each row lists an adapter considered by the router, its DP training certificate $(\varepsilon_{\text{train}}, \delta_{\text{train}})$, how many times the budget-aware controller selected it in each routing mode (Exact/Surrogate/Fallback) and in total, and the average per-hop inference privacy ε actually consumed. Under our post-processing posture, inference ε=0 indicates no additional DP noise at inference; training ε is incurred offline. Counts are aggregated over our reproducibility runs and will vary with seeds.

Adapter	Tier	DP cert	$\varepsilon_{\text{train}}$	δ_{train}	Chosen (runs)				Inference ε (avg)
					Exact	Surrogate	Fallback	Total	
LoRA-A-small	Small	certA	2.0	1×10^{-6}	240	0	0	240	0.0
LoRA-B-medium	Medium	certB	3.5	1×10^{-6}	0	680	0	680	0.0
LoRA-C-large	Large	certC	6.0	1×10^{-6}	0	0	0	0	0.0
(no-adapter)	None	—	—	—	0	0	402	402	0.0