Counterfactual Regret for Reflective Agents with Lean-Verified Guarantees

Anonymous authors
Paper under double-blind review

Abstract

Reflective language agents reuse feedback and memory to adapt across tasks without updating model weights. However, existing guarantees for policy selection and action generation do not by themselves establish when local reflection yields long-term performance guarantees for the complete agent under resource constraints. In this paper, we address this gap by deriving sufficient conditions for sublinear counterfactual regret from local generation and deployment guarantees, while retaining the native controller. Specifically, we establish a resource-sensitive transfer that bounds full-state deployment discrepancies along actual execution paths and preserves the own-history laws of both learner and comparator. Besides, local generation reliability yields a cumulative generation opportunity bound, linking policy acquisition to long-term performance through an explicit quality, coverage, and hard-budget tradeoff. To verify these guarantees, we formalize the conditional proof chain for all thirteen numbered theoretical results in Lean, including a native instance with repeated policy growth and imperfect validation. The complete Lean proof artifacts are available in an anonymous repository[1] for verification and reproduction.

1 Introduction

Reflective language agents use feedback to revise decisions and retain experience for subsequent tasks. With frozen model weights, linguistic reflection and memory provide a practical means of adaptation during execution. Reflexion and Self-Refine demonstrate benefits in coding, reasoning, and iterative task solving (Shinn et al., 2023; Madaan et al., 2023). Reusable instructions can carry lessons from earlier outcomes into later decisions, making reflection valuable for systems that must adapt through interaction. These capabilities motivate understanding how local feedback-driven changes contribute to long-term agent performance.

However, existing guarantees for policy selection and action generation do not by themselves establish sustained performance gains for the complete reflective process. With frozen model weights, useful reflection must produce reusable policies and influence subsequent execution within available resources. These effects are coupled because current decisions shape future feedback and policy availability. Understanding this coupling requires explicit conditions linking local adaptation to long-term performance. We study this gap through counterfactual regret against resource-feasible procedures on their own histories, following the policy-regret comparison principle (Arora et al., 2012). Rather than requiring every reflection to improve performance, we identify generation and deployment conditions under which average excess loss vanishes while retaining the native controller.

Contributions. We develop a conditional theory connecting local reflection mechanisms to a complete native-agent regret guarantee. Theorem 11 retains the native controller and identifies conditions for sublinear regret through three contributions.

1. *Resource-sensitive native transfer.* Full-state deployment discrepancies are bounded along actual resource paths, preserving both counterfactual marginals and connecting hard budgets to cumulative loss.
2. *Locally derived generation bounds.* Local reliability connects policy acquisition to long-term performance, with an explicit quality, coverage and hard-budget tradeoff.

[1] https://anonymous.4open.science/r/lean-proofs-78F0

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

3. *A Lean-verified conditional proof chain.* All thirteen numbered results are formalized, including a concrete native instance that derives compatible interfaces and a sublinear bound from its mechanisms.

Related work. Growing-expert tracking provides guarantees for expanding pools (Mourtada & Maillard, 2017), while generative action sets jointly optimize creation and selection through a context-conditioned oracle and quadratic mismatch model (Xu et al., 2026). We connect local generation reliability to complete reflective executions through a generation opportunity bound. Its composition with deployment stability captures how obtaining useful policies and executing them jointly affect regret.

Policy regret formalizes action-responsive comparisons (Arora et al., 2012), with recent Markov-game guarantees under function-approximation conditions (Nguyen-Tang & Arora, 2025). We develop a resource-sensitive transfer for a retained native controller. Full-state perturbations accumulate along actual execution paths, with hard budgets controlling their cumulative effect. The construction preserves both native marginals while connecting local deployment conditions to own-history regret.

Merten et al. (2018) established Coq-verified multiplicative weights and composed distributed-system guarantees. Our Lean development connects growing selection, local generation reliability, deployment stability, and completed-task correspondence through explicit interfaces, supporting reproducible checking of the composed no-regret guarantee.

2 Model and reference algorithm

We define the complete reflective execution before introducing a reference process whose selection guarantee can be transferred to it.

2.1 Histories, policies, and counterfactual regret

Use task indices $t = 0, \dots, T-1$, internal-operation indices $s \geq 0$, and reference-slot indices $r = 0, \dots, R$. These clocks differ. All logarithms are natural, $\mathbf{1}\{E\}$ is the indicator of event E, and $\langle u, v \rangle = \sum_i u(i)v(i)$. Appendix K collects notation and its Lean correspondence.

Definition 1 (Legal history and operations)**.** Fix measurable operation and result spaces U, O, an initial-history set $\mathcal{H}_0^{\mathrm{op}}$, and a measurable admissibility relation $\mathsf{E}(h, u, o)$. Define $\mathcal{H}_{s+1}^{\mathrm{op}} = \{h \,\|\, (u, o) \mid h \in \mathcal{H}_s^{\mathrm{op}},\ \mathsf{E}(h, u, o)\}$ and $\mathcal{H}^{\mathrm{op}} = \bigsqcup_{s \geq 0} \mathcal{H}_s^{\mathrm{op}}$, assuming each level measurable. Here $\|$ appends a completed operation and result. A measurable legal-operation set $\mathsf{U}(h)$ restricts the next control. Histories record contexts, primitive actions, structured outcomes, memory, versioned registrations, and resource counters. Execution can name only an already registered executable identity. Legal prefixes need not have positive probability. Hard budgets impose equation 2.

A history records what has happened. An execution policy specifies the distribution of the next primitive action, given the current context and memory.

Definition 2 (Execution policy and card identity)**.** Let $\mathsf{X}, \mathsf{M}, \mathsf{A}$ be measurable context, memory, and primitive-action spaces. Identity i names an immutable card program and interpreter version, inducing a probability kernel $\pi_i(da \mid x, m)$ on A. For every measurable $D \subseteq \mathsf{A}$, $\pi_i(D \mid x, m)$ is measurable in (x, m), and $\pi_i(\mathsf{A} \mid x, m) = 1$. Selecting i chooses a kernel, whereas sampling a executes an action. Memory changes may alter its output law without changing identity. Program or interpreter changes require a new identity.

A meta-policy governs the complete procedure, including when to create cards and which existing card to execute. It is therefore more than a selector over a fixed library.

Definition 3 (Meta-policy and own-history law)**.** A meta-policy $\mu(du \mid h)$ is a measurable probability kernel on operations with $\mu(\mathsf{U}(h) \mid h) = 1$. Generation, validation, memory updates, and execution are distinct operations. The history index allows time-dependent control. Composing this kernel with environment and runtime kernels, assuming internal operations terminate, induces task records $Z_t \in \mathcal{Z}_t$. Here Z_0 is the prescribed initial record and Z_{t+1} completes task t. The task history is $H_t = (Z_0, \dots, Z_t) \in \mathcal{H}_t^{\mathrm{task}} = \prod_{u=0}^{t} \mathcal{Z}_u$. Its recursively generated law is $\mathbb{P}_\mu$. A nonempty class $\mathcal{M}$ contains complete procedures, not distributions over current cards. The native learner A is another procedure and need not belong to $\mathcal{M}$.

The *native controller* is the operation-selection kernel $\mu_A(du \mid h)$ of procedure A, with $\mu_A(\mathsf{U}(h) \mid h) = 1$. Operation-result, environment, and runtime kernels additionally determine $\mathbb{P}_A$. Retaining this controller means that it need not execute the reference algorithm. Decisions may change across histories, and transfer conditions do not hold automatically.

Fix a common measurable loss scorer σ_t on $\mathcal{H}_{t+1}^{\text{task}}$ and set $Y_t = \sigma_t(H_{t+1}) \in [0, 1]$, with smaller values preferred. This scalar evaluates the structured output, rather than replacing it. Define

$$L_T^A = \sum_{t<T} Y_t^A, \qquad J_T(A) = \mathbb{E}_{\mathbb{P}_A} L_T^A, \qquad R_T(A, \mathcal{M}) = J_T(A) - \inf_{\mu \in \mathcal{M}} J_T(\mu). \tag{1}$$

The expectation includes environment, generation, validation, runtime, logging, and private control randomness present in the specified execution. A deterministic environment is permitted. The native model may respond causally to past actions. Oblivious loss tables are required only in the benchmark of Section 4.

For a measurable trajectory map r_A and source law Γ, $((r_A)_{\#}\Gamma)(D) = \Gamma(r_A^{-1}(D))$ for every measurable event D. The subscript # denotes *measure pushforward*, so $(r_A)_{\#}\Gamma = \mathbb{P}_A$ fixes the native trajectory law.

Proposition 4 (Well-defined native objective)**.** *The loss in equation 1 is measurable and integrable, with $0 \le J_T(A) \le T$. A measurable realization r_A on a coupling space with $(r_A)_{\#}\Gamma = \mathbb{P}_A$ satisfies $\mathbb{E}_\Gamma[L_T^A \circ r_A] = J_T(A)$.*

Proposition 4 preserves the full native trajectory law, not merely the current action distribution. This is required for every comparator.

The counterfactual is a complete execution. Both procedures use the same environment mechanism and scoring function, but supply their own preceding actions and memories to that mechanism. Equal initial conditions do not mean equal subsequent observations. Randomness may be coupled only with the correct marginals. In particular, the comparator cannot use a card solely because the learner generated it. The class restriction is fixed before taking the infimum. This is the distinction between comparison on a realized loss sequence and policy-regret comparison in a responsive system (Arora et al., 2012; Anava et al., 2015).

2.2 Resource constraints and interfaces

Require a nonempty comparator class satisfying, under each own law,

$$\mathbb{P}_\mu\{\text{Cost}_t^\mu \le B_t \text{ for every } t = 0, \dots, T\} = 1. \tag{2}$$

Here Cost_t^μ is cumulative resource use after t completed tasks, including failed internal attempts, and B_t is a deterministic prefix budget. Any additional operationwise limit belongs to $\mathsf{U}(h)$. Hard feasibility restricts procedures, not task loss. Mean feasibility instead bounds $\mathbb{E}_{\mathbb{P}_\mu} \text{Cost}_t^\mu$, assuming integrability, and is not equivalent to equation 2.

The *resource interface* determines admissible procedures. The *representation interface* maps a comparator to a causal execution-card path and charges any lost advantage. The *deployment interface* compares that represented execution with what the native runtime actually does. These maps need not commute. For example, compilation may consume resources, and imperfect implementation may change future memory. We therefore impose them in a specified order and account for adjacent differences, rather than treating them as independent interchangeable assumptions.

2.3 Growing-library exponential weights

The reference algorithm is a mathematical comparison process, not an installed controller. Write $\mathcal{F}_r^-$ for completed pre-draw information, with realization h_r. The nonempty finite identity set $\mathcal{I}_r = \mathcal{I}_r(h_r)$ and sampling law are measurable with respect to this information. Let $w_0 = \nu_0$ be a probability distribution and Θ_r a nonnegative row-stochastic transition from $\mathcal{I}_r$ to $\mathcal{I}_{r+1}$. For nonnegative estimated losses $\widehat{\ell}_r(i)$ and learning rate $\eta > 0$, define

$$p_r(i) = \frac{w_r(i)}{\sum_j w_r(j)}, \qquad w_{r+1}(j) = \sum_i w_r(i) e^{-\eta \widehat{\ell}_r(i)} \Theta_r(i, j). \tag{3}$$

This recurrence uses the EXP3 and Fixed-Share primitives (Auer et al., 2002; Herbster & Warmuth, 1998; Cesa-Bianchi & Lugosi, 2006). Its normalizers are positive. For persistent identity labels, Fixed Share uses

$$\Theta_r(i,j) = (1-\alpha)\mathbf{1}\{i=j\} + \alpha\nu_{r+1}(j), \qquad 0<\alpha<1, \tag{4}$$

where ν_{r+1} has full support and persistent labels share one immutable namespace. With a full-support initial prior, this specialization gives positive draw probabilities, including for newborn cards. A positive normalizer alone does not imply coordinatewise support. Generation is not optimized by this recurrence.

Algorithm 1 Growing-Library Exponential Weights

Require Slot cap $N = R+1$, $\eta > 0$, prior ν_0, and admissible run protocol
Require Accepted-score interface supplying M_r, y_r, q_r and positive sampling support
1. Initialize registered identities $\mathcal{I}_0$ and weights $w_0 \leftarrow \nu_0$.
2. **for** $r = 0, \dots, N-1$ **do**
3. **if** slot r is active according to completed history h_r **then**
4. Set $p_r(i) \leftarrow w_r(i)/\sum_j w_r(j)$ and sample $J_r \sim p_r$.
5. Execute the admissible scored run and receive the selected outcome O_r.
6. Set $\widehat{\ell}_r$ by equation 6 and settle task and resource costs.
7. **else**
8. Set $\widehat{\ell}_r \leftarrow 0$ and apply the terminal convention of Section 2.4.
9. **end if**
10. Form $\mathcal{I}_{r+1}$ from completed registrations and choose row-stochastic Θ_r.
11. Set $w_{r+1}(j) \leftarrow \sum_{i\in\mathcal{I}_r} w_r(i)e^{-\eta\widehat{\ell}_r(i)}\Theta_r(i,j)$.
12. **end for**

The protocol supplies lawful run boundaries and acceptance probabilities, not unexecuted losses. A valid arrival closes a scored slot before the next draw. New identities enter only after generation and validation, and can be selected at that next draw. There is no additional draw outside the loop. The terminal weight update is for analysis, not another task. The synchronized run contract in Section 2.4 remains essential.

2.4 Feedback model and selection guarantee

At fixed h_r, let $k_r(i,o)$ be the finite outcome law under reference execution of identity i, and let $\lambda_r(i,o)$ be its prescribed task-loss total. The target $\ell_r^{\mathrm{ref}}(i) = \sum_o k_r(i,o)\lambda_r(i,o)$ is defined by that execution, not by accepted logs. Let $M_r(i,o) \in \{0,1\}$ mark acceptance and $y_r(i,o) \in [0,d_r]$ be the proxy score, with predictable cap d_r. Suppressing h_r, require

$$q_r(i) = \sum_o k_r(i,o)M_r(i,o) > 0, \qquad \sum_o k_r(i,o)M_r(i,o)y_r(i,o) = q_r(i)\ell_r^{\mathrm{ref}}(i). \tag{5}$$

It is stronger than positive logging probability. It excludes selection-biased accepted scores unless an additional correction is proved. The estimator is

$$\widehat{\ell}_r(i) = \frac{\mathbf{1}\{J_r=i\}M_r(J_r,O_r)y_r(J_r,O_r)}{p_r(i)q_r(i)}. \tag{6}$$

Only the executed observation enters the update. A structured outcome is compatible with equation 5, because its score is a measurable function. The scalar score does not replace the full state used by reflection.

Covered stopping rules. Fix a deterministic cap $N = R+1$. The event r active means that the reference process has not terminated and completed history h_r authorizes slot r. Its indicator $\mathbf{1}\{r \text{ active}\}$ is 1 on this event and 0 otherwise, determined in $\mathcal{F}_r^-$ before the current draw. This is distinct from card availability $\mathcal{I}_r$ and feedback acceptance M_r. Inactive slots produce an absorbing terminal token and zero targets and estimates. A result may stop later slots, never retroactively cancel its own slot. Held-run length and eligibility require the synchronized accepted-score contract. Arbitrary current-draw-dependent aggregation needs an additional bridge, not merely a shared clock.

Lemma 5 (Local and stopped observation moments)**.** *In a finite observation model satisfying equation 5, with positive $p_r(i)$, the conditional draw–outcome law gives*

$$\mathbb{E}_{J_r,O_r|h_r}\widehat{\ell}_r(i) = \ell_r^{\mathrm{ref}}(i), \qquad \mathbb{E}_{J_r,O_r|h_r}\langle p_r, \widehat{\ell}_r^2\rangle \le d_r^2 \sum_{i\in\mathcal{I}_r} \frac{1}{q_r(i)}. \tag{7}$$

For a predictable comparator coordinate $v_r(h_r)$, let $e_r = \langle p_r, \widehat{\ell}_r\rangle - \widehat{\ell}_r(v_r) - \ell_r^{\mathrm{ref}}(J_r) + \ell_r^{\mathrm{ref}}(v_r)$. Under the recursively generated bounded stopped law $\mathbb{P}_{\mathrm{stop}}$,

$$\mathbb{E}_{\mathbb{P}_{\mathrm{stop}}} \sum_{r<N} \mathbf{1}\{r \text{ active}\} e_r = 0, \tag{8}$$

$$\mathbb{E}_{\mathbb{P}_{\mathrm{stop}}} \sum_{r<N} \mathbf{1}\{r \text{ active}\}\langle p_r, \widehat{\ell}_r^2\rangle \le \sum_{r<N} \mathbb{E}_{\mathbb{P}_{\mathrm{stop}}^{(r)}}\left[\mathbf{1}\{r \text{ active}\} d_r^2 \sum_i q_r(i)^{-1}\right]. \tag{9}$$

Here $\mathbb{P}_{\mathrm{stop}}^{(r)}$ is the actual completed-prefix law, not a distribution obtained by revealing the stopping time in advance.

From the potential to expected selection regret. A reference experiment supplies equation 3 and its integrable moments on one coupling Γ_μ. For a supported predictable path $\mathbf{v} = \mathbf{v}^\mu = (v_0, \dots, v_R)$ with $v_r \in \mathcal{I}_r$, define

$$C(\mathbf{v}) = -\log \nu_0(v_0) - \sum_{r<R} \log \Theta_r(v_r, v_{r+1}). \tag{10}$$

Require $C(\mathbf{v}) \le C$ on every modeled path, with deterministic C, and the deterministic second-moment budget

$$\sum_{r=0}^{R} \mathbb{E}_{\Gamma_\mu}\left[\sum_{i\in\mathcal{I}_r} p_r(i)\widehat{\ell}_r(i)^2\right] \le V_{\mathrm{est}}. \tag{11}$$

This is a raw second moment, not a centered variance. The coupling must induce the same pre-draw law as the observation interface. Path support alone does not ensure predictability.

Theorem 6 (Growing selection)**.** *Under these conditions,*

$$\sum_{r=0}^{R} \mathbb{E}_{\Gamma_\mu}[\ell_r^{\mathrm{ref}}(J_r) - \ell_r^{\mathrm{ref}}(v_r)] \le \frac{C}{\eta} + \frac{\eta V_{\mathrm{est}}}{2}. \tag{12}$$

Theorem 6 balances path complexity C against feedback uncertainty V_{est} through η. Its potential proof uses Lemma 5, with the observation interface and recurrence sharing the same pre-draw law. Equality of isolated identity marginals is insufficient. No-regret behavior is derived when these terms grow slowly enough.

3 Counterfactual regret guarantees

We first decompose regret, then bound deployment and generation effects locally. This order identifies which discrepancies each theorem controls.

3.1 Regret decomposition and transfer

Fix μ and a coupling Γ_μ preserving both native laws. Every endpoint below is an expected total at fixed horizon T, whose subscript is suppressed. In particular, $J^{\mathrm{sel}} = \sum_{r=0}^{R} \mathbb{E}_{\Gamma_\mu}\ell_r^{\mathrm{ref}}(J_r)$ and $J_\mu^{\mathrm{av}} = \sum_{r=0}^{R} \mathbb{E}_{\Gamma_\mu}\ell_r^{\mathrm{ref}}(v_r)$, whereas $J_A^{\mathrm{nat}} = J_T(A)$ and $J_\mu^{\mathrm{nat}} = J_T(\mu)$. Specify intermediate score random variables and integrate them under this same coupling. The resulting certificates are

$$\begin{aligned}
J_A^{\mathrm{nat}} - J_A^{\mathrm{ref}} &\le E_A, & J_A^{\mathrm{ref}} - J^{\mathrm{sel}} &\le F + \mathcal{P}_b + D_A,\\
J_\mu^{\mathrm{av}} - J_\mu^{\mathrm{al}} &\le G, & J_\mu^{\mathrm{al}} - J_\mu^{\mathrm{cl}} &\le bS,\\
J_\mu^{\mathrm{cl}} - J_\mu^{\mathrm{path}} &\le h_{\mathrm{resp}}(S+1) + D_C, & J_\mu^{\mathrm{path}} - J_\mu^{\mathrm{ref}} &\le E_{\mathrm{nf}},\\
J_\mu^{\mathrm{ref}} - J_\mu^{\mathrm{nat}} &\le E_C, & V_{\mathrm{est}} &\le bV,
\end{aligned} \tag{13}$$

where $\mathcal{P}_b = h_{\text{resp}}(\lceil T/b \rceil + N_b + 1)$. The integers $b \geq 1$ and $N_b, S, h_{\text{resp}} \geq 0$ are the nominal run cap, extra-boundary envelope, path switch bound, and response scale. A *certificate* here is an explicit bound such as equation 13, assumed at an interface or derived from local conditions, not an empirical validation. The response scale is not an automatic mixing time. The deterministic V bounds scaled second-moment capacity, not a Bellman value.

The learner reference score is J_A^{ref}. The available substitute has score J_μ^{av}. The score J_μ^{al} evaluates the ideal represented comparator path after alignment to run starts, on clean reference slots. The score J_μ^{cl} removes run alignment, J_μ^{path} removes the reference response discrepancy, and J_μ^{ref} is the reference meta-policy endpoint after undoing representation, before native deployment. These names specify the order of the comparison. Each endpoint must be defined before a certificate is asserted.

F charges displaced foreground task opportunities. $\mathcal{P}_b$ charges clean-prefix loss due to at most N_b extra boundaries. D_A, D_C are score-calibration allowances. G charges policies missing from the available library. S bounds represented path changes. E_{nf} charges the loss of advantage under representation. E_A, E_C charge implementation differences. Hard resource costs constrain admissibility separately. A birth is not automatically both a displaced task and a unit task penalty.

Theorem 7 (Conditional native counterfactual transfer)**.** *Let $\mathcal{M}$ be nonempty. Suppose every $\mu \in \mathcal{M}$ has a compatible reference experiment, own-law endpoint realizations, and certificates equation 13 with common budgets, C, and $\eta > 0$. Put $\Delta = E_A + F + D_A + G + D_C + E_{\text{nf}} + E_C$. Then*

$$R_T(A, \mathcal{M}) \leq U_T = \Delta + \frac{C}{\eta} + \frac{\eta b V}{2} + h_{\text{resp}}(\lceil T/b \rceil + N_b + 1) + bS + h_{\text{resp}}(S + 1). \tag{14}$$

Theorem 7 transfers a reference selection bound by accounting for each adjacent discrepancy once. The native learner need not execute Fixed Share, but complete-execution identities must be justified. Comparator couplings may differ while preserving the same learner law. Unrelated logs do not establish this correspondence.

Corollary 8 (Conditional no regret)**.** *Along a positive-horizon family, all budgets, $b, N_b, S, h_{\text{resp}}, C, V, \eta$ may depend on T but are uniform over μ at that horizon. If each of the five normalized terms*

$$\frac{\Delta}{T}, \quad \frac{C/\eta + \eta b V/2}{T}, \quad \frac{h_{\text{resp}}(\lceil T/b \rceil + N_b + 1)}{T}, \quad \frac{bS}{T}, \quad \frac{h_{\text{resp}}(S+1)}{T} \tag{15}$$

tends to zero, then $\max\{0, R_T(A, \mathcal{M})/T\} \to 0$.

This is upper no regret, not convergence of signed regret to zero. For fixed $h_{\text{resp}} > 0$ and fixed b, the third term has limit at least $h_{\text{resp}}/b > 0$. Thus fixed run caps do not automatically give sublinearity. Appendix K illustrates compatible horizon scalings.

3.2 Resource-sensitive deployment stability

Mean-preserving rounding need not preserve nonlinear reflection, so we compare full-state transition kernels. Here S_t ranges over a time-varying finite type containing all behavior-relevant memory, card-library, resource, and runtime components. Use common initial law p^{init} and scores $c_t(S_t) \in [0, 1]$. An update affects the next state, not the score already produced at boundary t. These are state-score positions. Appendix L proves the completed-task padding with $T + 1$ positions for T tasks and zero terminal resource, preserving both boundary scores.

Let K_t be a native kernel and L_t its feedback-processed reference. Define $J_T(K) = \mathbb{E}_{\mathbb{P}_K} \sum_{t<T} c_t(S_t)$. Assume

$$\text{TV}(K_t(s, \cdot), L_t(s, \cdot)) = \tfrac{1}{2} \sum_j |K_t(s, j) - L_t(s, j)| \leq \varepsilon_t(s). \tag{16}$$

For each remaining horizon n with $t + 1 + n \leq T$, let $V^L_{t+1,n}$ be its expected accumulated score. Whenever $\varepsilon_t(s) > 0$, assume its range over next states is contained in an interval of length $h_t(s) \geq 0$. At zero-error states no such range hypothesis is needed.

Theorem 9 (Local stability on the actual native path)**.** *The preceding local conditions imply*

$$|J_T(K) - J_T(L)| \leq \mathbb{E}_{\mathbb{P}_K} \sum_{t<T} h_t(S_t)\varepsilon_t(S_t). \tag{17}$$

For nonnegative $\bar{H}, \bar{\varepsilon}, \chi_t(s), r_t(s)$, assume $h_t(s) \le \bar{H}$, $\varepsilon_t(s) \le \bar{\varepsilon}\,\chi_t(s)$, and $c_{\min}\chi_t(s) \le r_t(s)$ for $c_{\min} > 0$. If every positive-probability native path has $\sum_{t<T} r_t(S_t) \le B_T$, then

$$|J_T(K) - J_T(L)| \le \bar{H}\bar{\varepsilon}B_T/c_{\min}. \tag{18}$$

Alternatively, without a bounded-influence assumption, a uniform local TV bound $\varepsilon_t(s) \le \bar{\varepsilon}_T$ gives $|J_T(K) - J_T(L)| \le T^2\bar{\varepsilon}_T$.

Theorem 9 weights each local shift by its downstream sensitivity along the actual law. Appendix J explains its role in joint resource design.

Theorem 10 (Resource-sensitive native transfer)**.** *Suppose Theorem 7 applies to the reference endpoints L_A, L_μ, with completed-task scores connected by Appendix L. Apply the local resource contract of Theorem 9 to K_A, L_A and uniformly to K_μ, L_μ, with budgets B_A, B_C and common $\bar{H}, \bar{\varepsilon}, c_{\min}$. Then*

$$J_T(K_A) - \inf_\mu J_T(K_\mu) \le U_T + \frac{\bar{H}\bar{\varepsilon}}{c_{\min}}(B_A + B_C). \tag{19}$$

For fixed $\bar{H}\bar{\varepsilon}/c_{\min}$, $U_T = o(T)$ and $B_A + B_C = o(T)$ imply upper no regret. The alternative local-only bound is sublinear if $T\bar{\varepsilon}_T \to 0$ on both endpoints.

Theorem 10 restores both native endpoints, hence the two resource budgets. These outer connections must not duplicate implementation allowances already comparing the same endpoints in equation 13.

3.3 Generation reliability and the main guarantee

Theorem 10 can be instantiated without assuming that the total generation gap is small. We call the following joint success and quality requirements *local generation reliability*. Fix each comparator before play. Decompose its available-to-aligned gap into finitely many causal generation episodes. In episode e, a normalized finite kernel generates trial outcomes using previous trial history. A Boolean analysis predicate identifies a validated and retained comparator-suitable outcome. On every still-unsuccessful history, its next-trial probability is at least $\rho_e \in (0, 1]$. The algorithm need not know which card is optimal. This is a conditional coverage assumption, not an oracle input or an independence assumption.

After k_e trials, let $\xi_e \in \{0, 1\}$ indicate that none succeeded under its actual episode law $\mathbb{P}_e$. There are τ_e delay tasks and n_e post-generation tasks. On each completed generation history, require the conditional expected per-task available-minus-aligned gap to be at most 1 during delay and at most $\delta_e + \xi_e$ afterwards, with $\delta_e \ge 0$. The successful card must support a causal available path satisfying this local quality condition. A useful text that is discarded or inaccessible is not a success. The episode sums must equal the actual generation endpoint difference, without counting the same task twice.

Theorem 11 (Native guarantee from local conditions)**.** *Write $g_T^{\mathrm{gen}}(\mu) = J_\mu^{\mathrm{av}} - J_\mu^{\mathrm{al}}$. Under these episode conditions, uniformly for the specified comparator class,*

$$g_T^{\mathrm{gen}}(\mu) \le \sum_e \{\tau_e + n_e[\delta_e + (1 - \rho_e)^{k_e}]\}. \tag{20}$$

For nonnegative envelopes, if $\sum_e \tau_e \le A_T$, $\sum_e n_e \le T$, $\delta_e \le \delta_T$, and $(1 - \rho_e)^{k_e} \le f_T$, the right side is at most $A_T + T(\delta_T + f_T)$. Keep the other reference certificates, use identity deployment arrows inside the reference transfer, and impose the full-state local resource contracts on the two outer native-to-reference arrows. Use the validated completed-task instance and padded stability clock of Appendix L. With U_T^{rest} denoting equation 14 with $G = E_A = E_C = 0$, this gives

$$R_T \le U_T^{\mathrm{rest}} + A_T + T(\delta_T + f_T) + \kappa(B_A + B_C), \qquad \kappa = \bar{H}\bar{\varepsilon}/c_{\min}. \tag{21}$$

For a positive-horizon family with fixed $\rho \in (0, 1]$ and κ, suppose $f_T = (1 - \rho)^{k_T}$, $k_T \to \infty$, $\delta_T \to 0$, and $U_T^{\mathrm{rest}}, A_T, B_A + B_C = o(T)$. Then $\max\{0, R_T/T\} \to 0$.

A concrete native instance. Appendix I constructs a native verified-template reflector satisfying the interfaces jointly. Appendix N extends the instance evidence to repeated growth with imperfect validation and persistent selection errors.

Corollary 12 (Native verified-template reflection)**.** *For each integer $k \geq 1$, the constructed instance has $T = k^2 + k$ tasks, hard budget $2k+2$, and own-history comparator class $\mathcal{M}_0$ containing immediate repair and no reflection. Then*

$$R_T(A_k, \mathcal{M}_0) \leq (2 + \log 2)k + k^2 2^{-k} + \tfrac{1}{4} \leq (3 + \log 2)k + \tfrac{1}{4}. \tag{22}$$

Positive normalized regret vanishes along this family, whereas no reflection has regret T.

Corollary 13 (Quality and hard-budget tradeoff)**.** *Fix a causal design satisfying the episode conditions and remaining interfaces of Theorem 7, uniformly over hard-feasible comparators. At tolerance $\delta_e \geq 0$ and coverage $\rho_e(\delta_e) \in (0, 1]$, let $c_e(\delta_e) \geq 0$ bound prepaid trial cost including failures, k_e bound prefix counts, and $B_T^{\text{other}} \geq 0$ bound other costs. Put $\overline{U}_T = U_T^{\text{rest}} + E_A + E_C$. If equation 23 holds, the learner is hard-feasible and satisfies equation 24, where*

$$B_T^{\text{other}} + \sum_e k_e c_e(\delta_e) \leq B_T, \tag{23}$$

$$R_T \leq \overline{U}_T + \sum_e \left\{ \tau_e + n_e \left[\delta_e + e^{-k_e \rho_e(\delta_e)} \right] \right\}. \tag{24}$$

For a positive-horizon family, if $\sum_e n_e \leq T$, $\delta_e \leq \delta_T \to 0$, $k_e \rho_e(\delta_e) \geq \lambda_T \to \infty$, and $\overline{U}_T + \sum_e \tau_e = o(T)$, then $\max\{0, R_T/T\} \to 0$.

Appendix J gives feasible quotas and compatibility with Corollary 12, charging delay and deployment once.

4 Near-matching bounds under paid growth

4.1 The paid-growth model

This theoretical benchmark isolates paid public growth and sparse feedback. It is not an empirical evaluation or a lower bound for every native interface. There are $m \geq 1$ public phases of $n \geq 1$ primitive rounds, with $T = mn$ and integer registration budget $B \geq 4m$. A public registration schedule releases a strictly growing library. With zero-based phase index $j < m$, set

$$K_j = \lceil B/2 \rceil + \left\lfloor \frac{(j+1)\lfloor B/2 \rfloor}{m} \right\rfloor. \tag{25}$$

Each registration costs one unit, the total is B, and all intermediate prefixes are hard-feasible. Old identities persist, unborn identities are illegal, and the schedule is public.

Put $j(t) = \lfloor t/n \rfloor$ and $[K]_0 = \{0, \ldots, K-1\}$. The legal table domain is $\mathcal{D} = \{(t, i) \mid 0 \leq t < T,\ i \in [K_{j(t)}]_0\}$. An oblivious table $\ell \in \{0,1\}^{\mathcal{D}}$ has entries $\ell_t^{\text{base}}(i)$. The learner emits $I_t \in [K_{j(t)}]_0$. Only its selected bit is observed, independently erased with probability $1 - q$, where $0 < q \leq 1$. Task loss is

$$Y_t = \tfrac{1}{2}\left(\ell_t^{\text{base}}(I_t) + \mathbf{1}\{t > 0,\ I_t \neq I_{t-1}\}\right). \tag{26}$$

The comparator path class is $\mathcal{V} = \prod_{j=0}^{m-1} [K_j]_0$. For $\mathbf{v} \in \mathcal{V}$, define $I_t^{\mathbf{v}} = v_{j(t)}$ and $L_T(\mathbf{v}, \ell)$ by summing equation 26 with $I_t = I_t^{\mathbf{v}}$. Thus both sides pay actual switches, the first draw is free, and registration is not charged again.

Let $\mathfrak{A}$ contain all causal randomized phase controllers, using all previously visible phase and primitive history. Define

$$\mathcal{R}_{\text{bin}}^*(T, m, B, q) = \inf_{A \in \mathfrak{A}} \sup_{\ell \in \{0,1\}^{\mathcal{D}}} \left\{ \mathbb{E}_{\mathbb{P}_{A,\ell,q}} \sum_{t<T} Y_t - \min_{\mathbf{v} \in \mathcal{V}} L_T(\mathbf{v}, \ell) \right\}, \tag{27}$$

The law $\mathbb{P}_{A,\ell,q}$ covers learner and logging randomness for fixed ℓ. Both bounds use this same model and cost.

Theorem 14 (Paid sparse-feedback benchmark)**.** *Suppose additionally that*

$$q^3 T \leq mB \log B. \tag{28}$$

Then

$$\frac{1}{128} \min\left\{ T, \sqrt{\frac{mBT}{q}} \right\} - \frac{m}{2} \leq \mathcal{R}_{\text{bin}}^* \leq \min\left\{ T,\ 8\sqrt{\frac{mB \log BT}{q}} + 5m \right\}. \tag{29}$$

The libraries grow strictly in every phase, total registration cost is B, and every intermediate registration prefix is hard-feasible.

4.2 Proof outline and sublinear regimes

The lower construction hides a good identity per phase. The upper learner counts actual switches. Appendix F proves matching up to logarithmic and boundary terms in this restricted regime.

Corollary 15 (Learnable sparse-feedback families)**.** *For benchmark instances satisfying equation 28, if*

$$\frac{mB\log B}{qT} \longrightarrow 0, \qquad \frac{m}{T} \longrightarrow 0, \tag{30}$$

then $\max\{0, \mathcal{R}^*_{\text{bin}}/T\} \to 0$.

These conditions force $q \to 0$ and require effective observations qT to dominate identification complexity. The benchmark and published all-q switching bounds (Dekel et al., 2014) have distinct scopes, detailed in Appendix F.

5 Lean formalization

Our formal contribution is a compositional verification of native reflection, from probabilistic execution and local contracts to counterfactual regret and sublinear limits. Table 1 identifies its verified connections.

Why composition requires verification. Valid component bounds yield a guarantee for the same agent only with compatible trajectory laws, scoring clocks and resource conventions. Our proofs verify marginal preservation, completed-task alignment and adaptive observation moments, establishing that the combined bound concerns the specified native execution.

Table 1: Proof obligations addressed by the Lean formalization.

Proof obligation	Purpose of formal checking	Formal result
Probability coherence	Preserve native laws across couplings.	Prop. 4, Thm. 7
Adaptive observations	Derive moments under bounded stopping.	Lem. 5, Thm. 6
Generation reliability	Couple quality, trials and hard costs.	Thm. 11, Cor. 13
Deployment stability	Control error propagation on actual paths.	Thms. 9–10
Composition and limits	Align task clocks and limit hypotheses.	Thm. 11, Cor. 8
Benchmark consistency	Match cost and feedback conventions.	Thm. 14
Concrete instantiation	Derive interfaces from native execution.	Cor. 12, Prop. 16

Reusable guarantees and scope. Concrete native instances derive compatible contracts from generation, validation and execution. Explicit interfaces identify which obligations must be re-established when mechanisms change, supporting reuse of the theory. All thirteen results have Lean proofs (de Moura & Ullrich, 2021; The mathlib Community, 2020) of the *end-to-end conditional mathematical implications*. Applications must establish their contracts. Appendix H supplies declaration mappings and build provenance for independent checking.

6 Conclusion

We connect local generation reliability and deployment stability to conditional native-agent regret, providing a unified view of how reflective mechanisms can affect long-term performance under explicit resource constraints. Concrete native instances demonstrate sublinear regret under hard budgets, repeated growth and imperfect validation. Lean verifies the composed proof chains, complemented by near-matching paid-growth bounds. The resulting framework separates algorithmic guarantees from application-specific assumptions through explicit interfaces, making clear which contracts must be established when generation, validation, or execution mechanisms change. This perspective enables reflective agents to be analyzed through modular guarantees, where improvements in individual components can be incorporated without changing the overall comparison framework. The results establish a foundation for reasoning about adaptive agents whose capabilities evolve through generation, validation, and execution.

Reproducibility statement

This paper reports mathematical results, not experiments. Appendix H maps all thirteen numbered results to Lean declarations. The anonymous companion includes proof modules, declaration logs, source hashes, and reproduction instructions. The import root includes the completed-task family proof used by Theorem 11 and the concrete native instance of Corollary 12. Reproduction uses Lean 4.19.0 and the recorded mathlib dependency. Local proof checking is distinct from a cold dependency-library rebuild. Appendices A–F give the original proofs, and Appendix I constructs the first instance. Appendix N adds repeated growth with imperfect validation. Prose-to-type correspondence and deployed implementation correctness are not claimed to be machine verified.

Ethics statement

No human-subject data or empirical agent measurements are used. Hard resources and runtime safety restrictions remain part of admissibility. A reference comparison does not authorize intervention in a deployed agent. Optimizing a task score does not justify bypassing safeguards.

AI use statement

Generative AI was used substantially for mathematical drafting, proof development, literature assistance, English writing, and document preparation. No empirical observations are reported. The submitting authors remain responsible for reviewing the claims, source correspondence, and final disclosure.

References

Noga Alon, Nicolò Cesa-Bianchi, Ofer Dekel, and Tomer Koren. Online learning with feedback graphs: Beyond bandits. In *Proceedings of the 28th Conference on Learning Theory*, volume 40, pp. 23–35, 2015. URL https://proceedings.mlr.press/v40/Alon15.html.

Oren Anava, Elad Hazan, and Shie Mannor. Online learning for adversaries with memory: Price of past mistakes. In *Advances in Neural Information Processing Systems*, volume 28, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/38913e1d6a7b94cb0f55994f679f5956-Abstract.html.

Raman Arora, Ofer Dekel, and Ambuj Tewari. Online bandit learning against an adaptive adversary: From regret to policy regret. In *Proceedings of the 29th International Conference on Machine Learning*, 2012. URL https://arxiv.org/abs/1206.6400.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002. URL https://epubs.siam.org/doi/10.1137/S0097539701398375.

Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012. URL https://arxiv.org/abs/1204.5721.

Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006. URL https://cesa-bianchi.di.unimi.it/predbook/.

Nicolò Cesa-Bianchi, Ofer Dekel, and Ohad Shamir. Online learning with switching costs and other adaptive adversaries. In *Advances in Neural Information Processing Systems*, volume 26, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/9cf81d8026a9018052c429cc4e56739b-Abstract.html.

Duo Cheng, Xingyu Zhou, and Bo Ji. Understanding the role of feedback in online learning with switching costs. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 5521–5543, 2023. URL https://proceedings.mlr.press/v202/cheng23f.html.

Amit Daniely, Alon Gonen, and Shai Shalev-Shwartz. Strongly adaptive online learning. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pp. 1405–1411, 2015. URL https://proceedings.mlr.press/v37/daniely15.html.

Leonardo de Moura and Sebastian Ullrich. The Lean 4 theorem prover and programming language. In *Automated Deduction—CADE 28*, pp. 625–635, 2021. URL https://lean-lang.org/papers/lean4.pdf.

Ofer Dekel, Jian Ding, Tomer Koren, and Yuval Peres. Bandits with switching costs: $T^{2/3}$ regret. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pp. 459–467, 2014. doi: 10.1145/2591796.2591868. URL https://arxiv.org/abs/1310.2997.

Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. In *Proceedings of the 28th International Conference on Machine Learning*, 2011. URL https://icml.cc/2011/papers/554_icmlpaper.pdf.

Ganesh Ghalme, Swapnil Dhamal, Shweta Jain, Sujit Gujar, and Yadati Narahari. Ballooning multi-armed bandits. *arXiv preprint arXiv:2001.10055*, 2020. URL https://arxiv.org/abs/2001.10055.

Mark Herbster and Manfred K. Warmuth. Tracking the best expert. *Machine Learning*, 32(2):151–178, 1998. URL https://link.springer.com/article/10.1023/A:1007424614876.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020. URL https://tor-lattimore.com/downloads/book/book.pdf.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Samuel Merten, Alexander Bagnall, and Gordon Stewart. Verified learning without regret: From algorithmic game theory to distributed systems with mechanized complexity guarantees. In *Programming Languages and Systems (ESOP 2018)*, volume 10801 of *Lecture Notes in Computer Science*, pp. 561–588, 2018. doi: 10.1007/978-3-319-89884-1_20. URL https://link.springer.com/chapter/10.1007/978-3-319-89884-1_20.

Jaouad Mourtada and Odalric-Ambrym Maillard. Efficient tracking of a growing number of experts. In *Proceedings of the 28th International Conference on Algorithmic Learning Theory*, volume 76, pp. 517–539, 2017. URL https://proceedings.mlr.press/v76/mourtada17a.html.

Quan M. Nguyen and Nishant Mehta. Near-optimal per-action regret bounds for sleeping bandits. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*, volume 238, pp. 2827–2835, 2024. URL https://proceedings.mlr.press/v238/nguyen24d.html.

Thanh Nguyen-Tang and Raman Arora. Policy-regret minimization in markov games with function approximation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 46242–46264, 2025. URL https://proceedings.mlr.press/v267/nguyen-tang25a.html.

Chanwoo Park, Xiangyu Liu, Asuman Ozdaglar, and Kaiqing Zhang. Do LLM agents have regret? a case study in online learning and games. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/65ea878cb90b440e8b4cd34fe0959914-Abstract-Conference.html.

Chloé Rouyer, Yevgeny Seldin, and Nicolò Cesa-Bianchi. An algorithm for stochastic and adversarial bandits with switching costs. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 9127–9135, 2021. URL https://proceedings.mlr.press/v139/rouyer21a.html.

Aadirupa Saha, Pierre Gaillard, and Michal Valko. Improved sleeping bandits with stochastic action sets and adversarial rewards. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 8357–8366, 2020. URL https://proceedings.mlr.press/v119/saha20a.html.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

The mathlib Community. The Lean mathematical library. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, 2020. doi: 10.1145/3372885.3373824. URL https://arxiv.org/abs/1910.09336.

Jianyu Xu, Vidhi Jain, Bryan Wilder, and Aarti Singh. Online decision making with generative action sets. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/07ed3b22786bd8d46a3654d5325bca09-Abstract-Conference.html.

Appendix

A Own-history executions and their realizations

A.1 Probability objects and proof of Proposition 4

Write $\mathcal{H}_t^{\text{task}}$ for the type of completed prefixes and $Q_t^A(h,\cdot)$ for the next-record kernel of A. Starting from its prescribed initial law $\mathbb{P}_A^{(0)}$, define the prefix law recursively by kernel integration. For a nonnegative measurable test function f this means

$$\int f(h,x)\, d\mathbb{P}_A^{(t+1)}(h,x) = \int_{\mathcal{H}_t^{\text{task}}} \left[\int f(h,x) Q_t^A(h,dx) \right] d\mathbb{P}_A^{(t)}(h). \tag{31}$$

Taking $f = 1$ and using $Q_t^A(h, \mathcal{Z}_{t+1}) = 1$ proves inductively that every prefix law is a probability measure. For a finite event model the inner integrals are ordinary finite sums. For the native measurable model they are the kernel integrals used in `NativeSemantics`. The comparator construction replaces Q^A by Q^μ everywhere, including its history argument.

Measurability and integrability. Let $\mathcal{H}_t^{\text{task}}$ denote the finite prefix of the product record space. A normalized measurable control kernel from $\mathcal{H}_t^{\text{task}}$ to $\mathcal{Z}_{t+1}$ recursively induces a trajectory probability law. In the companion, this is the causal trajectory construction used by `NativeSemantics.Execution.law`, not an unrelated measure supplied after the fact. Each score is a measurable function of a completed prefix. Composing with the measurable prefix projection makes Y_t measurable. Finite summation makes L_T measurable. Summing the unit bounds gives $0 \le L_T \le T$. Since the trajectory law is a probability measure, the constant T is integrable. Domination proves integrability of L_T. Monotonicity of integration gives $0 \le J_T \le T$.

For a measurable realization r_A, the pushforward identity means that its distribution is the actual causal law. Integration under a measurable pushforward gives

$$\int L_T^A(r_A(\omega))\, d\Gamma(\omega) = \int L_T^A(x)\, d((r_A)_\# \Gamma)(x) = \int L_T^A(x)\, d\mathbb{P}_A(x). \tag{32}$$

The preceding integrability argument justifies these real-valued integrals. The formal result is `NativeSemantics.Realization.expected_total`.

Finite loss bounds. For each trajectory x, the unit-score contract gives

$$0 \le \sum_{t=0}^{T-1} Y_t(x) \le \sum_{t=0}^{T-1} 1 = T. \tag{33}$$

Each summand is measurable, hence the finite sum is measurable. The integrable dominating constant has integral T because the law has total mass one. In particular, the comparator loss set is nonempty and bounded below by zero. Its real infimum is therefore the intended mathematical infimum, rather than a default value for an empty or unbounded set.

Why the comparator is outside the expectation. Fix a causal procedure μ before evaluating its expectation. Its initial conditions and kernels determine its own law $\mathbb{P}_\mu$. A coupling can correlate exogenous randomness across executions for a proof, but must preserve both marginals. It cannot give μ the library generated by A unless that library would also be generated in the comparator execution. In particular, an infimum over realized paths chosen after all random outcomes is not the infimum in equation 1.

Resource modes. The main theorem uses the hard class directly. If a mean class is considered, integrability of prefix costs is an additional prerequisite for ordinary real expectations. The checked inclusion theorem integrates each hard prefix inequality under the own law of the candidate. The hard-class optimum can exceed the mean-class optimum, so changing the comparator class requires changing the objective. It is not a modification of the implementation charge. These ancillary definitions and identities are in `ResourceClasses`. The revised main theorem does not assume the hard and mean classes coincide.

B Observation and stopping proofs

B.1 Local moments in Lemma 5

For this proof only, suppress the slot and history indices. Let I, O be finite types, $p(i) > 0$ with $\sum_i p(i) = 1$, and $k(i,o) \geq 0$ with $\sum_o k(i,o) = 1$. Let $M(i,o) \in \{0,1\}$ and $y(i,o) \in [0,d]$. Define

$$\widehat{\ell}_{j,o}(i) = \frac{\mathbf{1}\{j = i\} M(j,o) y(j,o)}{p(i)q(i)}, \qquad \mathbb{P}(J = j, O = o \mid h) = p(j)k(j,o). \tag{34}$$

All expectations in this subsection refer to this conditional joint law. In particular, the symbols in the estimator are evaluated at the observed event (j,o), whereas its coordinate is i.

First moment. Fix a completed history and suppress its index. The joint event probability of draw j and outcome o is $p(j)k(j,o)$. Its sum is one. At coordinate i, all summands with $j \neq i$ are zero. Hence

$$\sum_{j,o} p(j)k(j,o)\widehat{\ell}_{j,o}(i) = \frac{p(i)}{p(i)q(i)} \sum_o k(i,o)M(i,o)y(i,o) = \ell^{\mathrm{ref}}(i). \tag{35}$$

The division is legitimate because $p(i), q(i) > 0$. There is no hidden claim that a missing observation is itself a zero task loss. It contributes zero to the estimator and is corrected by the denominator.

Second moment. The accepted score satisfies $(My)^2 \leq Md^2$, because $M \in \{0,1\}$ and $0 \leq y \leq d$. Consequently

$$\mathbb{E}_{J,O|h}\widehat{\ell}(i)^2 \leq \frac{p(i)q(i)d^2}{p(i)^2 q(i)^2} = \frac{d^2}{p(i)q(i)}. \tag{36}$$

Multiplying by $p(i)$ and summing proves the second inequality in equation 7. Finite sums also give equality between the expected estimated learner–comparator difference and its true loss difference. The comparator coordinate must be fixed at this history. A coordinate selected using the current draw need not satisfy the same equality.

The learner–comparator moment identity. Let v be determined by the completed history, before (J,O) is drawn. Finite linearity and equation 35 give

$$\mathbb{E}_{J,O|h}\left[\sum_i p(i)\widehat{\ell}_{J,O}(i) - \widehat{\ell}_{J,O}(v)\right] = \sum_i p(i)\ell^{\mathrm{ref}}(i) - \ell^{\mathrm{ref}}(v), \tag{37}$$

$$\mathbb{E}_{J,O|h}[\ell^{\mathrm{ref}}(J) - \ell^{\mathrm{ref}}(v)] = \sum_i p(i)\ell^{\mathrm{ref}}(i) - \ell^{\mathrm{ref}}(v). \tag{38}$$

Subtracting the two equalities yields zero conditional estimation noise. No independence of the whole past and future is used. Predictability of v is essential because it permits applying the coordinate identity at the fixed coordinate v.

B.2 The generated stopped law

Let E_r be the finite event type at slot r. Complete histories are built recursively from Option(E_r). The terminal value `none` is absorbing. At an active history the next-event kernel is the normalized local observation kernel on `some` events. At an inactive history it assigns probability one to `none`. The Boolean activity rule depends only on the completed history. Let $\varpi_r(h)$ be the resulting prefix mass. Its recursion is

$$\varpi_0(()) = 1, \qquad \varpi_{r+1}(h,x) = \varpi_r(h)k_r^{\mathrm{stop}}(h,x). \tag{39}$$

Summing over x proves normalization by induction. Nonnegativity follows by multiplying nonnegative local masses. These are actual path probabilities, not importance weights.

For any cumulative event score F_r, expand one step of the expectation, sum over the last event, and use kernel normalization on the old score. This gives

$$\mathbb{E}_{\mathbb{P}^{(r+1)}_{\mathrm{stop}}} F_{r+1} = \mathbb{E}_{\mathbb{P}^{(r)}_{\mathrm{stop}}} F_r + \sum_h \varpi_r(h) \sum_x k_r^{\mathrm{stop}}(h,x) f_r(h,x). \tag{40}$$

For the estimation noise, the last sum is zero at inactive histories by definition and zero at active histories by equation 35. Induction proves equation 8. For the quadratic score, retain the actual local expectation instead of replacing it by zero, then apply equation 36. Summing the inequalities with nonnegative prefix masses proves equation 9. These final implications are `StoppedObservation.Protocol.stopped_unbiased` and `stopped_second`. They invoke the normalized recursive stopped process, rather than assuming a stopped martingale identity.

Detailed induction on the stopped prefix. Set $F_0 = 0$ and $F_{r+1}(h,x) = F_r(h) + f_r(h,x)$, with $f_r = 0$ on the terminal event. Expanding the finite product law gives

$$\sum_{h,x} \varpi_r(h) k_r^{\mathrm{stop}}(h,x) F_{r+1}(h,x) = \sum_h \varpi_r(h) F_r(h) \underbrace{\sum_x k_r^{\mathrm{stop}}(h,x)}_{=1} + \sum_h \varpi_r(h) \sum_x k_r^{\mathrm{stop}}(h,x) f_r(h,x). \tag{41}$$

For estimation noise, the second line vanishes at every history. The expectation remains zero at $r+1$ if it is zero at r, establishing the first stopped identity from the base case. For the quadratic score, it is bounded by

$$\sum_h \varpi_r(h) \mathbf{1}\{h \text{ is active}\}\, d_r(h)^2 \sum_{i \in \mathcal{I}_r} q_r(h,i)^{-1}. \tag{42}$$

Summing these one-step inequalities for $r = 0, \dots, N-1$ proves the cumulative capacity bound. Nonnegativity of ϖ_r preserves every inequality. Bounded N makes all sums finite, so no unbounded optional-stopping theorem is being silently applied.

Scope of the local modification. The observation kernel must represent the run that is actually being scored. If the chosen identity itself determines how many task rounds are included, inserting its original draw probability into a common-duration estimator is not justified by this proof. The revised statement retains the synchronized accepted-score contract at this precise boundary. The activity rule for the next completed slot remains fully history-adapted.

C Potential and native-transfer proofs

C.1 Proof of Theorem 6

Step 1. Positivity and conservation under sharing. The initial total mass is $W_0 = 1$. If $W_r > 0$, at least one $w_r(i)$ is positive. Since $\exp(-\eta \widehat{\ell}_r(i)) > 0$, the tilted mass is positive. Summing the update over j and using $\sum_j \Theta_r(i,j) = 1$ proves that sharing preserves this tilted total. Thus $W_{r+1} > 0$ by induction and all logarithms and probabilities below are defined.

Fix a realized sequence of finite types, transitions, and nonnegative estimates. Let $W_r = \sum_i w_r(i)$. Row stochasticity in equation 3 yields

$$W_{r+1} = W_r \sum_i p_r(i) e^{-\eta \widehat{\ell}_r(i)}. \tag{43}$$

The exponential is strictly positive. Starting from $W_0 = 1$, every W_r is positive. For $x \geq 0$, $e^{-x} \leq 1 - x + x^2/2$, and $\log u \leq u - 1$ for $u > 0$. Therefore

$$\log W_{r+1} - \log W_r \leq -\eta \langle p_r, \widehat{\ell}_r \rangle + \frac{\eta^2}{2} \langle p_r, \widehat{\ell}_r^2 \rangle. \tag{44}$$

No upper bound on the importance-weighted estimate is used here. This explains why arbitrarily small positive probabilities are compatible with the expectation proof, though not with an automatic high-probability statement.

Step 2. The logarithmic upper bound. For $x \geq 0$, $e^{-x} \leq 1-x+x^2/2$. Applying this coordinatewise and summing with probability weights gives

$$\frac{W_{r+1}}{W_r} \leq 1 - \eta \sum_i p_r(i)\widehat{\ell}_r(i) + \frac{\eta^2}{2} \sum_i p_r(i)\widehat{\ell}_r(i)^2, \tag{45}$$

$$\log W_{R+1} \leq -\eta \sum_{r=0}^{R} \langle p_r, \widehat{\ell}_r \rangle + \frac{\eta^2}{2} \sum_{r=0}^{R} \langle p_r, \widehat{\ell}_r^2 \rangle. \tag{46}$$

The first right side is positive because it bounds the strictly positive ratio from above. Applying $\log u \leq u-1$ to the ratio gives the one-step logarithmic inequality. The second display is its telescoping sum, using $\log W_0 = 0$.

Step 3. A supported path lower bound. Define its incoming mass at slot r by

$$\omega_r^{\mathbf{v}} = \nu_0(v_0) \prod_{s<r} \Theta_s(v_s, v_{s+1}) \exp\left(-\eta \sum_{s<r} \widehat{\ell}_s(v_s)\right). \tag{47}$$

Induction proves $w_r(v_r) \geq \omega_r^{\mathbf{v}}$. The base case is equality. For the induction step, retain the single nonnegative summand indexed by v_r in the update to $w_{r+1}(v_{r+1})$. Multiplication by the positive transition and exponential preserves the inequality. At the final slot, retain the mass at v_R and sum its outgoing row. This is why the final displayed bound uses R transitions but $R+1$ scored slots.

The supported path contributes a nonnegative term to the final total mass. Following it through slots $0, \ldots, R$ gives

$$W_{R+1} \geq \nu_0(v_0) \prod_{r<R} \Theta_r(v_r, v_{r+1}) \exp\left(-\eta \sum_{r=0}^{R} \widehat{\ell}_r(v_r)\right). \tag{48}$$

The last update is summed over its destination row, so there is no missing comparator transition after slot R. Taking logs, summing equation 44, and rearranging gives the deterministic path inequality

$$\sum_{r=0}^{R} \left(\langle p_r, \widehat{\ell}_r \rangle - \widehat{\ell}_r(v_r)\right) \leq \frac{C(\mathbf{v})}{\eta} + \frac{\eta}{2} \sum_{r=0}^{R} \langle p_r, \widehat{\ell}_r^2 \rangle. \tag{49}$$

Step 4. Combine the two sides and take expectation. The lower logarithm is $-C(\mathbf{v}) - \eta \sum_{r=0}^{R} \widehat{\ell}_r(v_r)$. Comparing it with equation 46, moving the loss terms to one side, and dividing by $\eta > 0$ proves equation 49. This argument is pointwise for every realized collection of finite types and stochastic transitions.

The proof is `Growing.growing_path_regret`. Integrate on the stipulated comparator coupling. The integrability assumptions permit finite summation and integration to commute. The moment identities identify the left side with actual reference selection differences. The code and quadratic budgets prove equation 12, as in `DependentSelection.expected_growing_selection`.

Compatibility with the observation theorem. To use the local calculation in the expected path theorem, the probabilities p_r in the event kernels must be those produced by the weight recurrence, and the score difference must be the same random variable. For history-dependent type families, these identifications and integrability are the explicit `Experiment` contract. The finite stopped-observation result proves moments for its stated event types. It does not construct every possible native observation model. Thus a supplied moment identity is a visible interface premise, not a claimed consequence of unrelated logs.

C.2 Proof of Theorem 7

For one fixed comparator, insert every endpoint in its stated order.

$$\begin{aligned} J_A^{\mathrm{nat}} - J_\mu^{\mathrm{nat}} = {} & (J_A^{\mathrm{nat}} - J_A^{\mathrm{ref}}) + (J_A^{\mathrm{ref}} - J^{\mathrm{sel}}) + (J^{\mathrm{sel}} - J_\mu^{\mathrm{av}}) + (J_\mu^{\mathrm{av}} - J_\mu^{\mathrm{al}}) \\ & + (J_\mu^{\mathrm{al}} - J_\mu^{\mathrm{cl}}) + (J_\mu^{\mathrm{cl}} - J_\mu^{\mathrm{path}}) + (J_\mu^{\mathrm{path}} - J_\mu^{\mathrm{ref}}) + (J_\mu^{\mathrm{ref}} - J_\mu^{\mathrm{nat}}). \end{aligned} \tag{50}$$

The selection arrow $J^{\mathrm{sel}} - J^{\mathrm{av}}_{\mu}$ is bounded by Theorem 6. Substitute all seven directed certificates and $V_{\mathrm{est}} \le bV$. This gives equation 14 for a fixed comparator. Proposition 4 identifies the realized native endpoint expectations with their own-law expected losses. The formal result permits comparator-specific coupling measures and type families.

Budget substitution without hidden sign assumptions. Substitution into equation 50 first gives

$$\begin{aligned} J_T(A) - J_T(\mu) \le E_A + (F + \mathcal{P}_b + D_A) + C/\eta + \eta V_{\mathrm{est}}/2 + G \\ + bS + h_{\mathrm{resp}}(S+1) + D_C + E_{\mathrm{nf}} + E_C. \end{aligned} \tag{51}$$

Since $\eta > 0$ and $V_{\mathrm{est}} \le bV$, replace only the quadratic term by $\eta bV/2$. Substitute $\mathcal{P}_b = h_{\mathrm{resp}}(\lceil T/b \rceil + N_b + 1)$ and group the seven scalar charges into Δ. This produces exactly U_T in equation 14. Each adjacent difference is present once. No absolute values or independently maximized versions of individual endpoints are inserted.

Write the uniform upper bound as U_T. For every μ, the fixed-comparator result states $J_T(A) - U_T \le J_T(\mu)$. Thus $J_T(A) - U_T$ is a lower bound on the nonempty set of comparator expected losses. The defining greatest-lower-bound property implies $J_T(A) - U_T \le \inf_{\mu} J_T(\mu)$, proving the theorem. There is no exchange of expectation and infimum.

Proof of Corollary 8. Divide the definition of U_T by its positive horizon. Its normalized value is exactly the sum in equation 15. Finite addition preserves limits, so $U_T/T \to 0$. The main bound implies

$$0 \le \max\{0, R_T/T\} \le \max\{0, U_T/T\}. \tag{52}$$

Continuity of the maximum and squeezing prove convergence. This is checked both as a numerical lemma and as a theorem on families of native executions with interface witnesses.

D Local-feedback stability and resource charging

Finite paths and Bellman values. For a kernel sequence K, define $V^K_{t,0}(s) = 0$ and

$$V^K_{t,n+1}(s) = c_t(s) + \sum_j K_t(s,j) V^K_{t+1,n}(j). \tag{53}$$

For an explicit next-state trace, its probability is the product of the traversed kernel probabilities. Inductively summing over the last or first next state proves normalization. Expanding the trace loss and using this normalization proves that the trace expectation equals equation 53. Averaging with the common initial distribution gives $J_T(K)$. These equalities are in `FeedbackTrajectory`, so the Bellman quantity is not merely a proxy with an unproved relation to the objective.

Range-sensitive TV inequality. If $f(j) \in [a, a+h]$ and p, q are probability distributions, their signed mass difference sums to zero. Subtract the midpoint $a + h/2$. Then

$$\left| \sum_j (p(j) - q(j)) f(j) \right| = \left| \sum_j (p(j) - q(j))(f(j) - a - h/2) \right| \le \tfrac{h}{2} \sum_j |p(j) - q(j)| = h\,\mathrm{TV}(p,q). \tag{54}$$

For zero TV the distributions agree on the finite set, so the expectation difference is zero without imposing a range condition.

D.1 Proof of Theorem 9

Step 1. Exact one-step difference. For a remaining horizon $n+1$, add and subtract the same reference continuation under K_t.

$$\begin{aligned} V^K_{t,n+1}(s) - V^L_{t,n+1}(s) = \sum_j K_t(s,j)\{V^K_{t+1,n}(j) - V^L_{t+1,n}(j)\} \\ + \sum_j \{K_t(s,j) - L_t(s,j)\} V^L_{t+1,n}(j). \end{aligned} \tag{55}$$

The common immediate score cancels. Taking absolute values uses the triangle inequality and $K_t(s, j) \geq 0$. The second sum is bounded by the range-sensitive TV inequality. If the local TV is zero, the distributions agree and this term is zero without a range condition.

Subtract the two Bellman equations and add and subtract the K-average of the reference continuation. The immediate scores cancel because they are the same function of the complete pre-update state. Writing $\zeta_t(s) = h_t(s)\varepsilon_t(s)$ gives

$$|V^K_{t,n+1}(s) - V^L_{t,n+1}(s)| \leq \sum_j K_t(s, j)|V^K_{t+1,n}(j) - V^L_{t+1,n}(j)| + \zeta_t(s). \tag{56}$$

The second term follows from equation 54 and the local contract. Induction on n, starting at zero continuation, bounds the right side by the K-Bellman value of the accumulated charge ζ. Averaging at time zero and using the trace interpretation proves equation 17.

Step 2. Propagate local charges along the native law. Define $\mathcal{D}_{t,0}(s) = 0$ and

$$\mathcal{D}_{t,n+1}(s) = \zeta_t(s) + \sum_j K_t(s, j)\mathcal{D}_{t+1,n}(j). \tag{57}$$

At $n = 0$, both value differences are zero. If $|V^K_{t+1,n} - V^L_{t+1,n}| \leq \mathcal{D}_{t+1,n}$, nonnegativity of K_t and equation 56 imply the same bound at $(t, n + 1)$. Therefore $|V^K_{0,T} - V^L_{0,T}| \leq \mathcal{D}_{0,T}$. Averaging over the common initial distribution p^{init} yields

$$\begin{aligned} |J_T(K) - J_T(L)| &\leq \sum_s p^{\text{init}}(s)|V^K_{0,T}(s) - V^L_{0,T}(s)| \\ &\leq \sum_s p^{\text{init}}(s)\mathcal{D}_{0,T}(s) = \mathbb{E}_{\mathbb{P}_K} \sum_{t<T} \zeta_t(S_t). \end{aligned} \tag{58}$$

The final equality is the finite trajectory/Bellman identity, applied to the nonnegative score ζ instead of the task score c. It fixes the law as $\mathbb{P}_K$, not a worst-case union of possible update times.

Step 3. The hard-resource inequality. For the hard-resource version, the pointwise hypotheses yield

$$0 \leq \zeta_t(s) \leq \bar{H}\bar{\varepsilon}\chi_t(s) \leq \frac{\bar{H}\bar{\varepsilon}}{c_{\min}} r_t(s). \tag{59}$$

Monotonicity of the K-Bellman recursion transfers this inequality to path expectations. For every positive-mass trace the cumulative resource is at most B_T. Zero-mass traces contribute zero, irrespective of their values. Normalization therefore gives $\mathbb{E}_{\mathbb{P}_K} \sum_t r_t(S_t) \leq B_T$. Substitute in equation 17 to prove equation 18. No worst-case union over possibly active times is introduced.

For the local-only alternative, induction in equation 53 gives $0 \leq V^L_{t,n} \leq n$. At boundary t, the remaining continuation has at most $T - t - 1$ scores. Hence

$$|J_T(K) - J_T(L)| \leq \sum_{t<T} (T - t - 1)\bar{\varepsilon}_T \leq T^2\bar{\varepsilon}_T. \tag{60}$$

This argument does not assume mixing or forgetting. Its precision condition is correspondingly stronger.

Proof of Theorem 10. For each fixed comparator, insert the two reference expectations.

$$J_T(K_A) - J_T(K_\mu) = [J_T(K_A) - J_T(L_A)] + [J_T(L_A) - J_T(L_\mu)] + [J_T(L_\mu) - J_T(K_\mu)]. \tag{61}$$

Bound the first and last terms by equation 18, and invoke the proved reference transfer for the middle term. Uniformity in μ permits the same infimum argument used for Theorem 7. In Lean, `NativeRepairEndpoint.counterfactual` calls `ModelMain.main_fixed`, together with actual kernel-path bounds. It does not accept the desired reference regret conclusion as a new axiom. Its resource-side structure contains local TV and influence bounds, local cost domination, and a hard path budget.

For no regret, normalize equation 19. The sum of U_T/T, a fixed constant times B_A/T, and a fixed constant times B_C/T tends to zero. Apply equation 52. For equation 60, division by T gives $T\bar{\varepsilon}_T$, whose assumed convergence proves the alternative.

E Local generation and the combined native theorem

Finite causal trial laws. Fix episode e and suppress e on its trial quantities. Let $\mathcal{H}^{\mathrm{gen}}_{e,k}$ be the finite type of completed trial outcomes in an episode. A normalized nonnegative kernel $g_k(h,i)$ selects the next outcome. Write $\mathbb{P}^{(k)}_e$ for the completed-trial law, with mass $\varpi^{\mathrm{gen}}_k(h)$. Its mass recursion satisfies $\varpi^{\mathrm{gen}}_0(()) = 1$ and $\varpi^{\mathrm{gen}}_{k+1}(h,i) = \varpi^{\mathrm{gen}}_k(h)g_k(h,i)$. The outcome can encode a candidate, validation, and retention. A success indicator $b_k(h,i) \in \{0,1\}$ marks an outcome whose retained card supports the stipulated causal available path. It may depend on the comparator fixed for the proof and on completed history, but is not an oracle supplied to the generator.

Let $\xi_0 = 1$ and $\xi_{k+1}(h,i) = \xi_k(h)(1 - b_k(h,i))$. On still-unsuccessful histories assume

$$\sum_i g_k(h,i)(1 - b_k(h,i)) \le 1 - \rho, \qquad 0 < \rho \le 1. \tag{62}$$

Expand one step of the actual law.

$$\mathbb{P}_e(\xi_{k+1} = 1) = \sum_h \varpi^{\mathrm{gen}}_k(h)\xi_k(h) \sum_i g_k(h,i)(1 - b_k(h,i)) \tag{63}$$

$$\le (1-\rho) \sum_h \varpi^{\mathrm{gen}}_k(h)\xi_k(h) = (1-\rho)\mathbb{P}_e(\xi_k = 1). \tag{64}$$

Successful-history terms vanish. All remaining path masses are nonnegative. Induction from probability one gives $\mathbb{P}_e(\xi_k = 1) \le (1-\rho)^k$, without trial independence. The checked result is `LocalGenerationCertificate.failure_bound`.

Local quality and the semantic endpoint. For completed trial history h, let $g^{\mathrm{pre}}_t(h)$ and $g^{\mathrm{post}}_t(h)$ be conditional expected task differences between a specified causal available path and its aligned ideal path. Their exact finite-law expectation must equal the corresponding part of $J^{\mathrm{av}}_\mu - J^{\mathrm{al}}_\mu$. This is a semantic identification, not a bound on the generation opportunity gap. An application must derive it from the score laws and a disjoint episode decomposition.

Assume $g^{\mathrm{pre}}_t(h) \le 1$ on τ delay tasks and $g^{\mathrm{post}}_t(h) \le \delta + \xi_k(h)$ on n later tasks. Then

$$\mathbb{E}_{\mathbb{P}^{(k)}_e}\left[\sum_{t<\tau} g^{\mathrm{pre}}_t + \sum_{t<n} g^{\mathrm{post}}_t\right] \le \mathbb{E}_{\mathbb{P}^{(k)}_e}[\tau + n\delta + n\xi_k] \tag{65}$$

$$\le \tau + n[\delta + (1-\rho)^k]. \tag{66}$$

The first inequality sums local bounds with nonnegative masses. Normalization evaluates constants. The second uses the proved failure bound. This is `Episode.gap_le`, not a field assuming a small total gap.

Episodes need not be independent, but their laws must describe their actual contributions. Random entry histories require uniformly valid conditional contracts and the correct marginal or conditional representation. Resetting history in the proof while keeping its effects in the deployment is invalid. Sum equation 66 to obtain equation 20. Nonnegative envelopes and $\sum_e n_e \le T$ give $A_T + T(\delta_T + f_T)$ by finite summation and monotonicity.

Uniform envelopes without separating an infimum. The comparison is first proved for a single fixed μ, with its own episode laws. For each episode, $n_e \ge 0$ and the envelope bounds imply

$$\tau_e + n_e[\delta_e + (1-\rho_e)^{k_e}] \le \tau_e + n_e(\delta_T + f_T). \tag{67}$$

Summing, using $\sum_e \tau_e \le A_T$, $\sum_e n_e \le T$, and $\delta_T + f_T \ge 0$ gives

$$g^{\mathrm{gen}}_T(\mu) \le A_T + (\delta_T + f_T)\sum_e n_e \le A_T + T(\delta_T + f_T). \tag{68}$$

Uniformity means these same numeric envelopes work for every comparator, although its episode laws may differ. There is no step that independently optimizes one envelope for one comparator and another envelope for a different comparator.

The combined endpoint. The formal structure `Remainder` retains acquisition, alignment, response, representation, and capacity conditions. It has no generation or deployment bound. The constructor `Remainder.certify` derives generation from episodes, identifies the inner deployment endpoints, and assigns zero to their budgets. The theorem `native_local` invokes the actual reference theorem with this constructed witness and the two local kernel-resource contracts. Thus the outer native kernel comparison includes any selector discrepancy, not just changes to displayed feedback.

The validated `CompletedNativeV19.Instance.bound` carries the kernels, reference experiment, episodes, integrable score endpoints, and completed-task identities. Its generation budget is defined by delay, quality, and failure envelopes, while its inner reference deployment arrows are identities. Its conclusion is the original own-history regret. No bound on that regret or on either derived total gap is an input. Other reference interfaces remain explicit. Subtracting the generation budget from the reference bound leaves exactly U_T^{rest}, proving equation 21 without duplicate deployment charges.

Exact budget identity. Let U_T be the reference upper bound with its generation budget $G = A_T + T(\delta_T + f_T)$ and internal deployment budgets zero. By definition $U_T^{\mathrm{rest}} = U_T - G$. Consequently

$$\begin{aligned} R_T &\le U_T + \kappa B_A + \kappa B_C \\ &= U_T^{\mathrm{rest}} + A_T + T(\delta_T + f_T) + \kappa(B_A + B_C). \end{aligned} \tag{69}$$

The first line is an application of the proved transfer, with the generation certificate just constructed. The second is an exact scalar identity. No adjacent endpoint difference may be charged twice. The same causal mechanism can affect different adjacent differences, which must both be retained. An outer full-state charge already covers propagation of its own perturbation. Appendix L gives an explicit accounting example.

The family no-regret theorem. Each horizon supplies a complete instance of the stated model and contracts. Its finite state domains, trial-outcome domains, episode index set, comparator coupling laws, and reference identity domains may vary with the horizon. Uniformity of a budget is over comparators at that horizon, not over all horizons. The formal family uses a common ambient measurable space and comparator index type. These do not require identical kernels or identical finite domains across horizons.

Divide equation 21 by $T > 0$. With fixed $\rho \in (0, 1]$ and κ, its normalized upper bound is

$$\frac{U_T^{\mathrm{rest}}}{T} + \frac{A_T}{T} + \delta_T + (1-\rho)^{k_T} + \kappa \frac{B_A(T) + B_C(T)}{T}. \tag{70}$$

Because $0 \le 1 - \rho < 1$ and $k_T \to \infty$, the geometric term tends to zero. Each remaining term tends to zero under the hypotheses of Theorem 11. This argument uses the combined resource condition directly, without adding separate resource limits. The positive part is continuous, so equation 52 proves the claimed upper no-regret limit.

The publication endpoint is `CompletedNativeV19.variable_family`. For each family index it invokes the validated completed-task bound, then combines the geometric and normalized limits with denominator T, not $T + 1$. It does not take the desired regret bound as a premise. The earlier `LocalNativeFamily.no_regret` remains a fixed-domain family result with separately normalized resource assumptions. The earlier correspondence endpoint remains a state-score family result. The new wrapper supplies the completed-task, variable-domain and combined-resource correspondence without changing equation 21.

Strength and cost of the assumptions. The available path must be causal, not chosen retrospectively from favorable trial outcomes. Retention and comparator-relative quality must persist on the episode being charged. If increasingly precise good cards become rarer, their smaller hazards must be reflected in f_T. A fixed positive hazard cannot be presumed for many mutually exclusive targets. Attempts also consume actual generation and validation resources. Increasing their number is legitimate only when delay and hard budgets permit it. These restrict the admissible native model class, not the original architecture. They are not proved properties of every deployed reflector.

F The finite paid lower and upper benchmark

The sparse-family conditions force $q \to 0$, not unlearnability at fixed q. This benchmark does not lower-bound every native certificate. Published all-q switching constructions (Dekel et al., 2014) address a broader regime.

F.1 Information lower bound for the full causal class

Finite information construction. In a phase with K identities, choose a hidden target uniformly. Its independent binary base losses have mean $1/2 - \epsilon$, while other identities have mean $1/2$. Let $\mathbb{P}_0$ be the all-$1/2$ observation law and $\mathbb{P}_j$ the target-j law for the same causal randomized learner. The erasure symbol is distinct from a zero loss. All laws are constructed from finite kernels on visible histories.

The Bernoulli divergence and erasure calculation give, for $0 \le \epsilon \le 1/4$,

$$D_{\mathrm{KL}}(\mathrm{Ber}(1/2) \| \mathrm{Ber}(1/2 - \epsilon)) = -\tfrac{1}{2}\log(1 - 4\epsilon^2) \le 4\epsilon^2. \tag{71}$$

Independent erasure multiplies this divergence by q. On each completed visible history the learner uses the same conditional action kernel under both environments. Its factor cancels in the supported finite chain rule. If N_j counts draws of j, induction on history length yields the divergence bound $4q\epsilon^2 \mathbb{E}_{\mathbb{P}_0} N_j$. Finite testing inequalities then give

$$\mathbb{E}_{\mathbb{P}_j} N_j \le \mathbb{E}_{\mathbb{P}_0} N_j + n\sqrt{4q\epsilon^2 \mathbb{E}_{\mathbb{P}_0} N_j}. \tag{72}$$

The divergence calculation in detail. For $u = 4\epsilon^2 \in [0, 1/4]$, the elementary inequality $-\log(1 - u) \le u/(1 - u)$ implies

$$-\tfrac{1}{2}\log(1 - 4\epsilon^2) \le \frac{2\epsilon^2}{1 - 4\epsilon^2} \le \tfrac{8}{3}\epsilon^2 \le 4\epsilon^2. \tag{73}$$

Let $E_q(p)$ be the distribution on $\{\perp, 0, 1\}$ assigning mass $1 - q$ to erasure and mass $qp(b)$ to bit b. The erasure contribution to relative entropy is zero and the factors q cancel in each non-erased likelihood ratio. Thus

$$D_{\mathrm{KL}}(E_q(p) \| E_q(p')) = \sum_{b \in \{0,1\}} qp(b) \log \frac{qp(b)}{qp'(b)} = qD_{\mathrm{KL}}(p \| p'). \tag{74}$$

At $q = 1$ the zero-mass erasure term is omitted. Positive Bernoulli probabilities on both sides ensure the same observation support.

At a fixed visible history h, write $p_t^{\mathrm{low}}(i \mid h)$ for the learner action kernel. This same function is used under $\mathbb{P}_0$ and $\mathbb{P}_j$, even though the distributions of h differ. On an action with positive mass, its factor cancels in the likelihood ratio. The next-event divergence is consequently

$$p_t^{\mathrm{low}}(j \mid h)\, q\, D_{\mathrm{KL}}(\mathrm{Ber}(1/2) \| \mathrm{Ber}(1/2 - \epsilon)). \tag{75}$$

For all other actions the observation laws agree. Induction with the supported finite chain rule adds these conditional divergences under the null prefix law. Since $N_j = \sum_{t<n} \mathbf{1}\{I_t = j\}$,

$$\begin{aligned} D_{\mathrm{KL}}(\mathbb{P}_0 \| \mathbb{P}_j) &\le 4q\epsilon^2 \sum_{t<n} \mathbb{E}_{\mathbb{P}_0^{(t)}} p_t^{\mathrm{low}}(j \mid H_t) \\ &= 4q\epsilon^2 \mathbb{E}_{\mathbb{P}_0} N_j. \end{aligned} \tag{76}$$

Finally $N_j \in [0, n]$, so its expectation difference is at most $n\,\mathrm{TV}(\mathbb{P}_0, \mathbb{P}_j)$. The included finite testing inequality bounds this TV by $\sqrt{D_{\mathrm{KL}}(\mathbb{P}_0 \| \mathbb{P}_j)}$, a conservative constant form. Combining with equation 76 proves equation 72. The proof handles zero-probability action histories on their support. It does not require the learner to assign positive probability to every action.

Write $x_j = \mathbb{E}_{\mathbb{P}_0} N_j$, $y_j = \mathbb{E}_{\mathbb{P}_j} N_j$. Since exactly one identity is selected per primitive round, $\sum_j x_j = n$. Cauchy–Schwarz bounds the sum of square roots in equation 72. Set

$$\epsilon = \min\{1/4, \sqrt{K/(64qn)}\}. \tag{77}$$

Then $64q\epsilon^2 n \le K$, and the calculation gives $K^{-1} \sum_j y_j \le 3n/4$. Expected regret against the hidden target is at least $\epsilon n/4$, which is at least $\min\{n, \sqrt{Kn/q}\}/32$. The scalar rate, the counts, and the underlying path distributions are all part of the proof chain, rather than a supplied information lower-bound certificate.

The count calculation and constants. Because $\sum_j x_j = n$, Cauchy–Schwarz gives $\sum_j \sqrt{x_j} \le \sqrt{Kn}$. Averaging equation 72 therefore gives

$$\frac{1}{K}\sum_j y_j \le \frac{n}{K} + n\sqrt{4q\epsilon^2 n/K} \le \frac{n}{2} + \frac{n}{4} = \frac{3n}{4}. \tag{78}$$

The second inequality uses $K \ge 2$ and $64q\epsilon^2 n \le K$. Against hidden target j, expected base regret is $\epsilon(n - y_j)$, because each non-target draw loses ϵ in conditional mean. Its average is at least $\epsilon n/4$. Substitution of equation 77 gives

$$\epsilon n/4 = \min\{n/16, \sqrt{Kn/q}/32\} \ge \tfrac{1}{32}\min\{n, \sqrt{Kn/q}\}. \tag{79}$$

The finite KL/testing lemmas used before this calculation belong to the included proof closure. Classical references describe the general information method (Lattimore & Szepesvári, 2020; Bubeck & Cesa-Bianchi, 2012), but no numerical lower bound from those references is inserted as an axiom.

Random tables and fixed instances. The loss table can be sampled before play. A finite factorization expresses the joint expectation as an expectation over complete oblivious tables followed by the learner and logging expectation conditional on each table. An average at least L implies at least one fixed table has conditional expected regret at least L. Independent phase targets allow the argument to be conditioned on every preceding visible phase transcript. Thus it applies to the full causal phase-controller class, not just to independently restarted learners.

For the paid schedule, each eligible set has size between $B/2$ and B. Adding phases, applying the half-scale task definition equation 26, and charging comparator boundary switches produces the left side of equation 29. The explicit constant $1/128$ is conservative. The proof uses the actual growing identity types in both the learner and the comparator, as recorded in `PaidInformationLower.paid_minimax_lower`.

F.2 The primitive upper algorithm and its tuning

Public construction. Within phase j, the public learner uses uniform initialization and a held-run cap and learning rate determined by K_j, n, q. Primitive execution records the identity emitted on every round. The proof identifies this primitive algorithm with its exponential-weights state recursion and counts the switches in the emitted identities. A partial final run is treated explicitly. No equality between a macro-policy and an unrelated primitive process is postulated.

For roundwise independent masks M_t, the run estimator for a fixed eligible identity is $\mathbf{1}\{J_r = i\}p_r(i)^{-1}\sum_{t\in\mathcal{T}_r} M_t \ell_t^{\text{base}}(i)/q$. Expanding its square separates the diagonal terms from the cross-time terms. Independence gives a diagonal bound proportional to u_r/q and a cross-time bound proportional to u_r^2. Applying exponential weights and charging actual held-run switches gives the phase bound. Phase composition, the actual boundary costs, and uniform library bounds give equation 86. These steps terminate at `PrimitiveJointBounds.uniform_minimax_upper`, in the same primitive minimax model as the lower bound.

Runwise variance, including cross-time terms. Here $\mathcal{T}_r$ is the consecutive task-index set in run r, $u_r = |\mathcal{T}_r|$, and u denotes one such length. In the following local calculation $s = 1, \dots, u$ numbers positions inside the run, not internal operations. The vector $\mathbf{M} = (M_1, \dots, M_u)$ contains its independent observation masks. The law $\mathbb{P}^{\text{phase}}_{A,\ell,q}$ below is the induced phase law of the specified upper learner, with the loss table fixed. The symbols $L^{\text{task}}_{\text{phase}}$ and $L^{\text{base}}_{\text{phase}}(i)$ denote its cumulative task loss and the cumulative base loss of fixed identity i in that phase. Fix the oblivious table and the history at the start of a held run of length u. Conditional on $J = i$, let $X_i = \sum_{s=1}^{u} M_s \ell_s^{\text{base}}(i)/q$. Independent Bernoulli masks give

$$\begin{aligned}\mathbb{E}_{\mathbf{M}|J=i,h}[X_i^2] &= \frac{1}{q}\sum_s \ell_s^{\text{base}}(i)^2 + 2\sum_{s<t}\ell_s^{\text{base}}(i)\ell_t^{\text{base}}(i)\\ &\le u/q + u(u-1) \le u/q + u^2.\end{aligned} \tag{80}$$

For $\widehat{\ell}(i) = \mathbf{1}\{J = i\}X_i/p(i)$, it follows that

$$\mathbb{E}_{J,M|h}\sum_i p(i)\widehat{\ell}(i)^2 \le K(u/q + u^2). \tag{81}$$

For run lengths bounded by b and totaling n, $\sum u^2 \le bn$ and $\sum u = n$. Thus the phase capacity is $V_{\text{phase}} = Kn(b + 1/q)$. In particular, the whole squared run loss is not multiplied by $1/q$. Such a replacement would conflate one observation per run with independent per-round erasure.

Exact public tuning. Write $a = K \log K$ and set

$$b = \min\{n, \max\{1, \lceil (n/a)^{1/3} \rceil\}\}, \qquad V_{\text{phase}} = Kn(b + 1/q), \qquad \eta = \sqrt{2 \log K / V_{\text{phase}}}. \tag{82}$$

These are the `TunedPhase` parameters. They depend on public n, K, q, not on the optimal arm or future losses. Since $n \ge 1$, $K \ge 2$, and $q > 0$, both b and η are positive. Holding an identity creates no within-run switch. Counting boundaries and the possible incomplete last run bounds internal switching by $n/b + 1$. The checked primitive-to-run correspondence then yields

$$\begin{aligned} \mathbb{E}_{\mathbb{P}^{\text{phase}}_{A,\ell,q}}[L^{\text{task}}_{\text{phase}}] - \tfrac{1}{2} L^{\text{base}}_{\text{phase}}(i) &\le \tfrac{1}{2}\{\log K/\eta + \eta V_{\text{phase}}/2 + n/b + 1\} \\ &= \tfrac{1}{2}\{\sqrt{2 \log K \, V_{\text{phase}}} + n/b + 1\}. \end{aligned} \tag{83}$$

The equality follows from $\eta^2 V_{\text{phase}} = 2 \log K$. Dividing by $2\eta > 0$ gives $\log K/\eta = \eta V_{\text{phase}}/2$, and both terms sum to $\eta V_{\text{phase}} = \sqrt{2 \log K \, V_{\text{phase}}}$.

Integer caps and short phases. The formal tuning proof treats the cap cases separately. If $a \ge n$, the capped run size is 1 and the phase task loss bound n is already at most $a^{1/3} n^{2/3}$. If $a < n$, put $x = (n/a)^{1/3} > 1$. Here $a \ge 1$ and $n \ge 1$ imply $x \le n$, so $x \le b \le x + 1 \le 2x$. Consequently

$$\sqrt{2anb} \le 2a^{1/3} n^{2/3}, \qquad n/b \le a^{1/3} n^{2/3}. \tag{84}$$

Also $\sqrt{2 \log K \, V_{\text{phase}}} \le \sqrt{2anb} + \sqrt{2an/q}$. Combining these estimates, or using the trivial bound in the short-phase case, gives the conservative phase bound $4\{a^{1/3} n^{2/3} + \sqrt{an/q} + 1\}$. The checked implementation includes its final partial run and actual emitted identities.

Phase composition and a single controller class. One additional boundary switch costs at most $1/2$ per phase. The upper controller resets its weights at the public phase boundaries, while the lower bound allows every causal controller to retain past history. Both controllers are nevertheless members of the same class $\mathfrak{A}$. Since $K_j \le B$ and $x \log x$ is increasing for $x \ge 1$, summing the phase bounds and using $T = mn$ gives

$$m(B \log B)^{1/3} n^{2/3} = (mB \log B)^{1/3} T^{2/3}, \qquad m\sqrt{B \log B \, n/q} = \sqrt{mB \log B \, T/q}. \tag{85}$$

The additive $4m + m/2$ is bounded by $5m$. Taking the supremum over fixed tables and then the infimum over controllers preserves this upper bound because the displayed public controller is admissible.

The preceding primitive argument gives the explicit bound

$$\mathcal{R}^*_{\text{bin}} \le 4\left((mB \log B)^{1/3} T^{2/3} + \sqrt{mB \log B \, T/q}\right) + 5m. \tag{86}$$

Proof of the sparse simplification. Let $x = a^{1/3} T^{2/3}$ and $y = \sqrt{aT/q}$, with $a, T, q > 0$. Then $x^3 = aT^2$ and $y^2 = aT/q$. Multiplying $q^3 T \le a$ by the nonnegative factor $a^2 T^3$ shows $x^6 \le y^6$. Both x and y are nonnegative, so monotonicity of the sixth power implies $x \le y$. Substitute into equation 86. Independently, cumulative task loss is at most T and the comparator task loss is nonnegative. The minimax value is therefore at most T. These two upper bounds prove the minimum in equation 29.

The registration assertions are not inferred from the loss bound. They are separate arithmetic theorems about equation 25. Every phase creates at least one identity under $B \ge 4m$, the increments telescope to the total budget, and each intermediate registration prefix is bounded by the phase endpoint. Their conjunction with the minimax statements is checked in `PrimitiveJointBounds.paid_common_model`.

Proof of Corollary 15. For positive T, q and nonnegative a,

$$\frac{\sqrt{aT/q}}{T} = \sqrt{\frac{a}{qT}}. \tag{87}$$

This follows by squaring both nonnegative sides, or by the square-root scaling identity. Take $a = mB \log B$. Continuity of the square root and equation 30 show that the normalized upper bound tends to zero. Apply equation 52. The proof in `SparseFeedbackBenchmark.sparse_no_regret` includes the normalization identity rather than treating asymptotic notation as an informal replacement for it.

G How to instantiate the native transfer

This section describes obligations for general applications. Appendix I separately discharges them for a specified verified-template reflector. Neither section reports measurements of a deployed language model. Consider a frozen-weight coding agent whose completed record contains the task, generated program, test report, memory, versioned rule cards, and remaining validation budget. A fixed scorer maps the structured test report to a value in $[0, 1]$. A rule card induces the execution policy π_i. The native controller μ_A is retained, rather than replaced by Algorithm 1.

State and identity. The state must retain all components needed to predict future task scores. A card identifier names an immutable program and interpreter version, not its current probability of selection. If a finite-state local theorem is used, a finite alphabet and bounded storage must be specified, including the environment state needed for the Markov representation. Bounded prompt length alone does not make an unbounded external environment finite-state. An abstraction that discards state requires its own justified comparison, not an assertion that the finite theorem applies unchanged.

Reference observation interface. An evaluator must identify a synchronized score unit, the draw probability $p_r(i)$, and the acceptance probability $q_r(i)$. Randomly accepting complete test reports independently of their score is one way to meet the accepted-score identity. An evaluator that retains only successful tests is not. A rule card generated after an outcome is available for the next draw, not for the draw that generated that outcome. Endogenous termination of future slots is allowed, whereas changing the current score window after seeing a favorable outcome requires a separate analysis.

Local feedback interface. Specify a native raw-feedback kernel K_t and a processed-feedback reference kernel L_t on the same complete state space. The required quantity is their total variation, including the joint distribution of generated cards, memory, resources, and runtime outcomes. Preserving the mean of the displayed score does not imply this condition. Theorem 9 requires a uniform continuation-value range at perturbed states. Such a range needs a mixing, reset, or bounded-influence argument on the complete state. Deleting a card after a fixed number of rounds does not prove that argument. If no such property is available, only the explicitly weaker local-only bound is supplied here.

Resources, generation, and selection are different obligations. Count failed generation and validation attempts whenever their resource is consumed. A strictly positive minimum resource cost controls the number of active updates on each native path. Theorem 11 derives the generation certificate from conditional coverage, per-task quality, and a causal available path. The number of registered cards alone proves none of these conditions. Finally, a native selection bridge must compare complete native and reference executions. An offline reference that receives logs from the native run is not automatically distributed as the online reference used in the potential argument. The learner law must stay fixed across comparator couplings.

Proved claims and required application evidence. The moment and potential bounds follow from the actual reference recursion and its stated local observation contract. The stability charge follows from complete-state TV, continuation range, and pathwise costs. The remaining directed certificates are assumptions to establish for the chosen model, preferably from local operational lemmas rather than from measured total regret. Finite observational data alone does not prove a uniform counterfactual certificate over all histories. A statistical validation theorem would need a specified sampling design, coverage assumptions, and uncertainty bounds. No such empirical certificate is claimed in this paper.

Table 2: Numbered results and their checked Lean endpoints. Prefix `DRGPR.` is omitted.

Result	Formal endpoint
Proposition 4	`NativeSemantics.Execution.expected_bounds`, `loss_integrable`, and `Realization.expected_total`
Lemma 5	`FiniteObservation.Round.estimate_mean`, `weighted_second`, and `StoppedObservation.Protocol.stopped_unbiased`, `stopped_second`
Theorem 6	`DependentSelection.expected_growing_selection`
Theorem 7	`NativeFamily.own_coupling_counterfactual`
Corollary 8	`NativeNoRegret.budget_tends_zero` and `NativeFamily.own_coupling_no_regret`
Theorem 9	`StateFeedbackStability.actual_loss`, `adaptive_resource_loss`, and module `FeedbackLocalOnly`, declaration `uniform_local_bound`
Theorem 10	`NativeRepairEndpoint.counterfactual`, `no_regret`, and `FeedbackLocalOnly.local_only_rate`
Theorem 11	`LocalGenerationCertificate.Episode.gap_le`, `uniform_budget`, `CompletedNativeV19.Instance.bound`, and `CompletedNativeV19.variable_family`
Corollary 12	Module `ConcreteReflection`, declarations `execution_identity`, `conditional_main`, `instantiated_bound`, `family_bound`, `square_rate`, and `no_regret`
Corollary 13	Module `JointGenerationBudget`, declarations `native_joint`, `hard_prefix`, `variable_family`, and `polynomial_allocation`
Theorem 14	`SparseFeedbackBenchmark.paid_sparse_capped` and `PrimitiveJointBounds.paid_common_model`
Corollary 15	`SparseFeedbackBenchmark.sparse_no_regret`
Proposition 16	Module `ImperfectGrowth`, declarations `pointer_error`, `actual_execution`, `actual_regret`, `full_transfer`, `actual_no_regret`, and `sublinear_resources`

No duplicate charging. Choose the reference endpoints before evaluating any error. A local-feedback charge can replace the corresponding deployment certificate, or can compare a new outer pair of executions as in Theorem 10. It cannot be added to an unchanged certificate for the same endpoint difference. Different adjacent differences may share a causal source. Resource feasibility restricts the comparator domain. Resource-induced displacement contributes to F only if it displaces a task opportunity not already included in another endpoint. The paid benchmark registers births under its hard budget and charges actual switches in its task loss. Neither cost is an additional generic penalty in the native objective.

H Declaration-level verification map

Definitions 1–3 and the thirteen theoretical results use a single numbering sequence. Definitions fix semantic objects and are not themselves proof claims. Table 2 records the checked endpoints. Intermediate declarations are reached through their Lean imports. Corollary 12 constructs a specified native model and derives its interfaces. Proposition 16 adds repeated native growth and derives the persistent cost of false acceptance. Other applications must still establish their own contracts. The release includes a machine-readable mapping with source and build hashes.

H.1 Meaning of end-to-end verification

The import root includes the correspondence, clock, validated-instance, observation-law and concrete-reflection modules. The publication wrapper scores completed tasks with the proved padding identity and requires integrable intermediate endpoints. Appendix L maps its supporting derivations. Exact declaration counts, source hashes and build provenance are recorded in the release manifest. These support the thirteen numbered paper results. Declaration count is not a measure of scientific novelty.

There are three distinct levels. At the declaration level, every numbered result maps to checked Lean declarations under their displayed hypotheses. At the conditional composition level, the combined endpoint invokes proved selection, generation, and native kernel-transfer results before taking the counterfactual infimum, and the family theorem invokes this endpoint before proving convergence. This is the mathematical chain certified by the artifact. At the concrete deployment level, one would additionally need a particular reflector, its scoring and logging implementation, and proofs that they instantiate every input contract. This third claim is not made.

In particular, the `Experiment` structure contains integrability, supported-path, moment, and capacity obligations. The local observation theorem proves moment identities for its specified finite interface. An instantiation must connect that interface to the same recurrence and probability law. Likewise, `Remainder` retains acquisition, alignment, history-response, representation, and capacity conditions. The new local theorem derives generation and deployment charges, not these remaining conditions. Compilation of an implication does not convert its assumptions into facts about a real model.

The companion contains exact source hashes, per-declaration axiom logs, and a dependency-closed source set. Only `propext`, `Classical.choice`, and `Quot.sound` are allowed. There is no `sorryAx` or custom regret axiom. English statements and their Lean types were compared manually. The Lean kernel and the recorded dependency environment are trusted, rather than verified by this paper. Citations document prior work, not a claim that all cited papers have been formalized.

Earlier broader claims are not verified. The previous all-q multiscale switching-rate claim is not a theorem of this version. Neither is a universal equivalence between raw-real native reflection and bit-feedback reflection, nor a universal validity claim for current-draw-dependent run stopping. The local restrictions in the revised statements are substantive and are recorded in the change audit. They are not represented as successful proofs of the broader earlier statements.

Position relative to related models. Tracking growing expert pools (Mourtada & Maillard, 2017), sleeping bandits (Nguyen & Mehta, 2024), and costly generative action sets (Xu et al., 2026) already address important forms of changing availability. We do not claim that growth, generation, or counterfactual response alone is new. The focus here is the formal transfer between complete executions and the local conditions under which its error can be controlled. These references provide context, not unproved imported premises of the numbered results.

I A FULLY SPECIFIED NATIVE REFLECTION INSTANCE

This appendix proves Corollary 12 without changing the general theorems. It provides a mathematical existence witness, not a measurement of a deployed language model. The instance is deliberately validation-dominant. A complete validator identifies a useful program, so the native controller can use it without learning an external selection rule. Generation can nevertheless fail, the registered library genuinely grows, deployment has a nonzero discrepancy, and all operation prefixes respect the same hard budget. The reference Bandit is used only in the proof. The construction does not establish joint tightness of all terms in the general bound.

Scope of the instance. The guarantee concerns the specified frozen template generator, complete validator, retention rule, native schedule, and localized deployment check. The no-regret conclusion is along a horizon-parameterized family, rather than a single anytime controller. Applying this design pattern to a language-model agent requires establishing the corresponding generation, validation, retention, resource, and deployment conditions for that agent. The example establishes joint feasibility of the native-transfer interfaces and a concrete benefit of reflection within this model.

I.1 TASKS, PROGRAMS, AND THE FROZEN REFLECTION MECHANISM

Fix integers $k, N \geq 1$ and $T = k + N$. Task t presents $x_t \in \{0, 1\}$ and requests the program output x_t. The context sequence can be fixed arbitrarily before execution. A structured result records the answer, program identity, validation status, and remaining resources. The common task scorer is zero for the correct answer and one otherwise. It is applied identically to the learner and every comparator.

There are three immutable program identities. Their executable content is

$$P_0(x) = x + 1, \qquad P_1(x) = x + 2, \qquad P_2(x) = x. \tag{88}$$

Initially only P_0, P_1 are registered. A complete validator tests both inputs, so it accepts P_2 and rejects the other two for this specification. This is possible because the template family and task domain are finite. We do not assume that finite tests establish correctness for arbitrary programs.

The fixed reflection kernel supports two prompts. A generic repair prompt returns P_2 or P_1 with equal probability. A guided template prompt returns P_2 deterministically. These are two specified input modes of the same unchanged kernel, available to both learner and comparator. The native learner uses the generic prompt. The comparator below uses the guided prompt. No learned weight, success probability, or hidden optimal-card label is supplied to the native controller.

After each of the first k completed tasks, the native learner makes one generic generation and validation attempt. Every successful candidate is registered immediately unless its immutable identity is already present. The controller retains the success flag and uses P_0 throughout this prescribed preparation phase. Thus early registration does not force immediate execution. After the kth attempt, it executes P_2 if present and P_0 otherwise. This batching choice incurs an explicit k-task acquisition cost. It is a behavior of this particular controller, not a restriction on the rapid-registration interface.

Ordinary execution and execution with an additional check are legal runtime operations for every procedure. On the first subsequent task only, the native schedule invokes the check, which drops a successful output with probability $1/4$, replacing it by the fallback answer. Its private bit is not retained in memory or the structured result. It does not modify registered programs, future tasks, or the future control rule. Later tasks execute the retained program normally. The clean reference performs the same charged check but leaves the output unchanged, so both resource counters agree. The native controller is consequently neither the reference Bandit nor identical to its own clean deployment.

I.2 Own-history comparators and prefix resources

The comparator class $\mathcal{M}_0$ contains two procedures. Procedure μ_0 always uses P_0. Procedure μ_1 makes one guided generation, validates and registers its output before the first task, and then uses P_2. Pre-task internal operations are legal for both procedures and the learner. The native learner simply follows its prescribed later schedule. The environment and scoring rule are unchanged. Consequently

$$J_T(\mu_0) = T, \qquad J_T(\mu_1) = 0, \qquad \inf_{\mu \in \mathcal{M}_0} J_T(\mu) = 0. \tag{89}$$

These are separate counterfactual executions, not comparator actions evaluated against the native memory. Nonnegative scores also imply that adding other legal comparators cannot lower this infimum below zero. The two-element class is sufficient for the formal witness.

Each attempt is charged two units at initiation, one for generation and one for its validation. Registration of the new identity consumes one further unit, and the deployment check consumes one unit. Initial registrations are part of the common initial condition. If r attempts have been initiated and $u, v \in \{0, 1\}$ indicate whether registration and the check have been charged, every native prefix satisfies

$$C_r = 2 \min\{r, k\} + u + v \le 2k + 2. \tag{90}$$

The guided comparator costs three units, at most $2k + 2$. The fallback comparator costs zero. Upfront charging also covers prefixes inside an unfinished attempt. Failed attempts are not refunded. Resource charges restrict admissibility and are not added again to the task scorer. The separate local deployment resource is one check, so its activity budget is $B_A = 1$, while $B_C = 0$.

I.3 The native probability law and actual task losses

Let $H_k = (\zeta_1, \dots, \zeta_k) \in \{0, 1\}^k$ be the generic trial outcomes, with success encoded by one. For a prefix h_j, the conditional next-outcome probability is $1/2$ for either value. The resulting finite law is the recursively normalized product law, not an assumed expected-loss formula. Define

$$\xi(h) = \mathbf{1}\{\zeta_1 = \cdots = \zeta_k = 0\}, \qquad \mathbb{P}(\xi(H_k) = 1) = 2^{-k}. \tag{91}$$

The deployment bit D has probabilities $\mathbb{P}(D = 1) = 1/4$ and $\mathbb{P}(D = 0) = 3/4$, independently of the trial history. Pre-sampling these bits is only a representation of the law. Decisions access each trial result after its generation and the deployment bit only at its check.

For each realized (h, d), the completed-task loss is

$$Y_t(h, d) = \begin{cases} 1, & 0 \le t < k, \\ \xi(h) + \mathbf{1}\{t = k,\ d = 1\}(1 - \xi(h)), & k \le t < T. \end{cases} \tag{92}$$

All scores lie in $[0, 1]$. Summing this actual process before taking expectations gives

$$\sum_{t<T} Y_t(h, d) = k + N\xi(h) + d(1 - \xi(h)), \qquad J_T(A) = \mathbb{E}_{H_k}\left[k + N\xi(H_k) + \tfrac{1}{4}(1 - \xi(H_k))\right]. \tag{93}$$

The Lean declarations `task_sum`, `conditional_execution`, and `execution_identity` establish this correspondence. Thus the expected loss is derived from task execution rather than introduced as an unconstrained endpoint.

I.4 Local generation and full-state deployment conditions

Each unsuccessful trial history has conditional success probability $\rho = 1/2$. A successful card is correct on every subsequent task and is retained. Therefore $\delta = 0$. The available post-preparation path uses P_2 when generated and a fallback otherwise. Its difference from the ideal comparator path is $N\xi(H_k)$. Applying the existing local-generation theorem gives

$$G = \mathbb{E}_{H_k}[N\xi(H_k)] \le N2^{-k}. \tag{94}$$

Here the episode delay is zero because the entire preparation loss is already charged as $F = k$. Charging another k in G would duplicate the same loss. The exact failure probability follows as well by induction on the independent binary trials.

At the deployment check, the future-relevant state consists of the retained success flag, the output-failure bit, and the deterministic task and resource counters. The registered library is determined by the success flag. Raw historical records can also be retained, but neither kernel uses the past output bit in a future decision. The native and clean kernels preserve the same memory and counters. Their complete stochastic state distributions have

$$\mathrm{TV}(K, L) = \tfrac{1}{4}(1 - \xi(h)). \tag{95}$$

After that task they induce identical future control and losses. Conditional on the completed prefix h, the retained memory is fixed and the future loss is common to both output branches. Their continuation-value span is at most one. The local discrepancy is at most $1/4$ and the check costs one unit. Averaging the resulting conditional difference, or directly using equation 93, gives the deployment certificate

$$E_A \le 1 \cdot \tfrac{1}{4} \cdot B_A = \tfrac{1}{4}, \qquad E_C = 0. \tag{96}$$

The formalization computes TV on the joint memory-output state, proves that later task scores ignore the slip bit, and establishes the exact endpoint difference. The span-one statement is conditional on the fixed memory, not a global span assertion across successful and unsuccessful memories. Those memories can have very different future values. This instance therefore supplies the deployment edge of Theorem 7 directly. It does not assume the stronger global range contract of Theorem 9 with $\bar{H} = 1$ on the unconditioned mixed-memory state space.

I.5 The reference Bandit and the native selection bridge

Condition on a completed generation history h, before drawing any reference action. Consider a two-arm reference pool. Arm zero is P_0. Arm one is P_2 if available, and otherwise P_1. These are distinct immutable identities in either case. The unused registered fallback need not enter this comparison pool. Registration is causal because the reference experiment begins after the k attempts. The native registered library grows on every successful trial trajectory, even though the subsequent comparison pool has two members.

The reference algorithm initializes both probabilities at $1/2$ and performs ordinary exponential weighting with learning rate $\eta > 0$ and identity transition. Only the selected score enters the update. Its score is

$$\ell_h(0) = 1, \qquad \ell_h(1) = \xi(h), \qquad q_r(i) = 1. \tag{97}$$

The observation kernel is deterministic conditional on the selected identity. It therefore satisfies the accepted-score identity exactly. Positivity of exponential weights gives full support at every finite time. The comparator path is arm one, with code $C = \log 2$ and no switches. The standard conditional second-moment calculation gives $V_{\mathrm{est}} \le 2N$. The constructed Lean `Program` recursively generates its draw probabilities and observations. Its capacity and code conditions are proved, not supplied as assumptions.

Let $J_{\mathrm{sel}}(h)$ be its expected post-preparation loss and $J_{\mathrm{av}}(h) = N\xi(h)$. Both arms have loss at least $\xi(h)$, so

$$N\xi(h) \le J_{\mathrm{sel}}(h), \qquad J_{\mathrm{sel}}(h) - N\xi(h) \le \frac{\log 2}{\eta} + \eta N. \tag{98}$$

The first inequality is the crucial native bridge. The clean native controller has total loss $k + N\xi(h)$, hence its acquisition edge is at most k. This is a one-sided performance comparison proved from validation, not an assertion that native sampling probabilities equal the Bandit probabilities. On a successful history, the native clean controller always uses the correct card. The initial reference draw uses the fallback with probability $1/2$, and the deployed native controller uses it with probability $1/4$ on its first post-preparation task. The controllers are genuinely distinct.

I.6 A causal coupling with the correct native marginal

Write $w_k(h)$ for the recursively generated trial probability, $s(d)$ for the deployment-bit probability and $Q_{\eta,h}(g)$ for the law of a complete N-slot reference trajectory. The latter law comes from the constructed Bandit recursion, not from native action logs. On the finite product space define

$$\Gamma(h, d, g) = w_k(h)s(d)Q_{\eta,h}(g), \qquad \sum_{h,d,g} \Gamma(h, d, g) = 1. \tag{99}$$

Each factor is nonnegative and normalized at its proper stage. The reference uses h only after all k trials have completed. It does not use the subsequent deployment bit. Thus this conditioning does not reveal future generation outcomes to an earlier draw.

The native output functional in equation 92 does not depend on g. Marginalizing the freshly generated reference path gives

$$\sum_{h,d,g} \Gamma(h, d, g) \sum_{t<T} Y_t(h, d) = \sum_{h,d} w_k(h)s(d) \sum_{t<T} Y_t(h, d) = J_T(A). \tag{100}$$

Both comparators have deterministic scores on their own executions, so they can be coupled to this space without changing either native law. The learner marginal is the same for both comparator choices. The Lean declarations `joint_law_normalized` and `native_marginal` prove these identities using normalization of the actual reference recursion. This supplies the compatibility missing from a comparison of isolated action marginals.

I.7 All certificates in one experiment

For the best comparator μ_1, set all ideal comparator-side endpoints to zero. This is its actual loss, represented exactly by P_2, whether or not that program is available to the learner. The available-to-ideal difference is precisely the generation edge. The fallback comparator has nonnegative loss, so the same final upper bound applies to it. No learner marginal depends on this comparator choice.

Use one task per reference slot, no response contamination and no path switches. In the notation of Theorem 7, the proved common budgets are

$$\begin{gathered} b = 1, \quad S = h_{\mathrm{resp}} = N_b = 0, \quad C = \log 2, \quad V = 2N, \\ F = k, \quad G = N2^{-k}, \quad E_A = \tfrac{1}{4}, \quad E_C = D_A = D_C = E_{\mathrm{nf}} = 0. \end{gathered} \tag{101}$$

Generation cost constrains equation 90. The preparation task loss appears only in F. Failed post-preparation opportunities appear only in G. The deployment check appears only in E_A. Exact

validation and the absence of an endogenous environment state make the remaining edges zero by construction, not by a generic assumption about reflective agents.

Apply the unchanged native-transfer theorem conditionally on h, using the actual reference recursion. Then average over the normalized generation law and use the local-generation theorem. Equation equation 89 identifies the own-history comparator infimum. This proves

$$R_T(A, \mathcal{M}_0) \le k + N2^{-k} + \tfrac{1}{4} + \frac{\log 2}{\eta} + \eta N. \tag{102}$$

In Lean, `conditional_main` invokes `ModelMain.finite_controller_transfer`. The declaration `instantiated_bound` averages that theorem and invokes `generation_bound`. The proof therefore uses the existing selection and transfer chain rather than merely proving a numerically similar inequality in isolation.

I.8 The horizon-indexed family and its scope

For $k = n + 1$, put $N = k^2$, $T = k^2 + k$ and $\eta = 1/k$. Equation equation 102 becomes the first inequality in Corollary 12. Induction gives $k \le 2^k$, hence $k^2 2^{-k} \le k$. Since $k^2 \le T$, this is an $O(\sqrt{T})$ bound along the specified horizons. More explicitly,

$$\frac{R_T(A, \mathcal{M}_0)}{T} \le \frac{2 + \log 2}{k} + 2^{-k} + \frac{1}{4k^2} \longrightarrow 0. \tag{103}$$

The positive-part limit is checked in Lean. Without reflection, μ_0 has regret exactly T, so the construction is not a comparison against the learner itself. It is an existence result for a restricted verified-template model, not an optimal-rate theorem, an arbitrary-language-model guarantee, or a single horizon-free controller. The constant coverage and complete validator are strong, explicit structural features. Removing them requires additional analysis. The model nevertheless demonstrates that genuine rule generation, retained native control, nonzero deployment error and hard resource feasibility can coexist with the unchanged theorem and a nonvacuous sublinear conclusion.

J Joint generation quality and hard resources

This appendix proves Corollary 13. The existing transfer theorem is unchanged. The new step couples the required policy quality to conditional coverage, trial quotas, and actual hard resources. A fixed design determines these parameters and a causal trial schedule. Selecting a design for a horizon does not replace the native controller by a Bandit controller. The guarantee applies when that native procedure satisfies the stated interfaces for the selected design.

J.1 Contracts and the meaning of the parameters

Fix a comparator before execution. For episode e, quality tolerance δ_e bounds the conditional expected available-minus-aligned task gap after success. A success means that a comparator-suitable rule has been generated, validated, retained, and made causally usable. It is an analysis predicate, not an optimal-policy label supplied to the native procedure. The lower bound $\rho_e(\delta_e) \in (0, 1]$ must hold after every still-unsuccessful trial history. Improving the requested quality may decrease this bound. A separate statement about text quality alone would not imply the task-gap contract.

The quota $k_e \in \mathbb{N}$ bounds attempted generation and validation operations. Successful episodes may stop early, with inactive trials padded for analysis. Such padding consumes no additional resource and cannot undo a success. The worst-case per-attempt envelope $c_e(\delta_e)$ includes failures. Costs outside this envelope, such as registration, are included in B_T^{other}. The plan is fixed before play, though its trial kernels and legal decisions may depend on completed history. Random redesign of the quotas requires a separate predictable-envelope argument.

Episode slots refer to one learner schedule shared by all comparator analyses. Success predicates may depend on the comparator, but the same physical attempt is charged once. Resource costs are not summed over counterfactual comparators. The common plan supplies uniform envelopes for their separate generation-gap proofs.

For a legal learner prefix h, let $u_e(h)$ be its initiated attempt count and $C^{\mathrm{other}}(h)$ its remaining charged costs. Assume each attempt reserves its cost before unfinished internal work can violate a budget, all costs are nonnegative, $u_e(h) \le k_e$, and $C^{\mathrm{other}}(h) \le B_T^{\mathrm{other}}$. Then

$$C_A(h) \le C^{\mathrm{other}}(h) + \sum_e u_e(h) c_e(\delta_e) \le B_T^{\mathrm{other}} + \sum_e k_e c_e(\delta_e) \le B_T. \tag{104}$$

This proves every-prefix hard feasibility, including prefixes inside a failed or unfinished attempt. It does not replace the comparator admissibility definition. Each comparator still executes on its own history and satisfies its own original hard constraint. In particular, a comparator cannot borrow a card or unused resource from the learner. These resource costs constrain execution rather than adding a second loss penalty.

J.2 Conditional failure, task gaps, and native regret

Let $F_{e,j}$ be the event that the first j trials in episode e have not succeeded, and let $\mathcal{F}_{e,j}$ record those completed trials. Under the actual episode law $\mathbb{P}_e$, conditional coverage implies

$$\mathbb{P}_e(F_{e,j+1}) = \mathbb{E}_{\mathbb{P}_e}\left[\mathbf{1}_{F_{e,j}} \mathbb{P}_e(F_{e,j+1} \mid \mathcal{F}_{e,j})\right] \le [1 - \rho_e(\delta_e)]\mathbb{P}_e(F_{e,j}). \tag{105}$$

The first equality uses that $F_{e,j+1} \subseteq F_{e,j}$ and the tower rule. On a successful history the indicator is zero. On an unsuccessful history the conditional failure probability is at most $1 - \rho_e(\delta_e)$. Starting from probability one and applying this argument k_e times gives

$$\mathbb{P}_e(F_{e,k_e}) \le [1 - \rho_e(\delta_e)]^{k_e} \le \exp\{-k_e \rho_e(\delta_e)\}. \tag{106}$$

For the second inequality, $1 - x \le e^{-x}$ for $x \in [0, 1]$, both sides are nonnegative, and raising to the nonnegative integer power preserves order. No independence assumption was introduced.

On every completed generation history, the conditional expected per-task gap is bounded by one during the τ_e delay tasks and by $\delta_e + \mathbf{1}_{F_{e,k_e}}$ for the n_e subsequent tasks. Integrating under the same episode law gives

$$\mathbb{E}_{\mathbb{P}_e}\left[\sum_{t<\tau_e} g_{e,t}^{\mathrm{pre}} + \sum_{t<n_e} g_{e,t}^{\mathrm{post}}\right] \le \tau_e + n_e\left[\delta_e + e^{-k_e \rho_e(\delta_e)}\right]. \tag{107}$$

The random variables $g_{e,t}^{\mathrm{pre}}$ and $g_{e,t}^{\mathrm{post}}$ are conditional task-gap functions on completed trial histories, as in Appendix E. Their sums must represent the actual available-to-aligned endpoint difference. Summing these identities and bounds establishes the generation certificate. Uniformity over the original comparator class permits the counterfactual infimum in Theorem 7. Its remaining certificates, reference selection theorem, and own-law realizations are unchanged. Substitution proves equation 24.

The quantity $\overline{U}_T$ contains all remaining reference and deployment terms. In particular, direct deployment certificates and those proved by Theorem 9 are both admissible. Replacing a direct certificate with a local proof does not create an additional copy of the same endpoint difference. Likewise, a preparation delay already charged in F has zero additional delay in equation 107.

In Theorem 9, local shift ε_t is weighted by downstream sensitivity h_t, along the actual law $\mathbb{P}_K$. Resource use limits the accumulated charge, not the union of all possible reflection times. Bounded card lifetime alone does not bound h_t, since descendants and environment changes can retain the effect.

For the asymptotic claim, the uniform envelopes in the corollary give

$$\frac{R_T}{T} \le \frac{\overline{U}_T + \sum_e \tau_e}{T} + \delta_T + e^{-\lambda_T}. \tag{108}$$

The right side tends to zero. Continuity of the positive-part map and the nonnegative lower bound on $\max\{0, R_T/T\}$ yield the claimed limit. This argument derives a regret bound from native execution witnesses before taking a limit. It does not assume that regret itself is sublinear. A vanishing coverage probability is allowed, provided cumulative exposure $k_e \rho_e(\delta_e)$ still diverges uniformly. Increasing k_e alone does not establish this property.

J.3 Explicit quotas and a polynomial quality-cost family

For a desired log-failure level $\Lambda > 0$, a coverage lower bound $\rho_\star > 0$, and per-attempt cost envelope $c_\star \geq 0$, define

$$k = \left\lceil \frac{\Lambda}{\rho_\star} \right\rceil, \qquad k\rho_\star \geq \Lambda, \qquad kc_\star \leq \left(\frac{\Lambda}{\rho_\star} + 1 \right) c_\star. \tag{109}$$

The first bound follows from $\lceil x \rceil \geq x$. The second follows from $\lceil x \rceil < x + 1$ for $x \geq 0$, weakening the strict inequality when $c_\star = 0$. Equations equation 106 and equation 109 bound the failure probability by $e^{-\Lambda}$. A design can therefore check hard feasibility using these explicit quotas, including the integer-rounding cost.

One illustrative class has constants $a_{\mathrm{g}} \in (0, 1]$, $c_0 > 0$, integer exponents $d_{\mathrm{g}}, d_{\mathrm{v}} \geq 0$ with $\nu = d_{\mathrm{g}} + d_{\mathrm{v}} > 0$, and, uniformly over episodes and comparators,

$$\rho_e(\delta) \geq a_{\mathrm{g}} \delta^{d_{\mathrm{g}}}, \qquad c_e(\delta) \leq c_0 \delta^{-d_{\mathrm{v}}}, \qquad 0 < \delta \leq 1. \tag{110}$$

The exponent d_{g} describes how quickly suitable candidates become rare as quality improves. The exponent d_{v} describes increasing generation or validation cost. These are additional properties of this illustrative class, not new assumptions imposed on every model in the main theorem. Integer exponents suffice for this formally checked example.

Choose $\rho_\star = a_{\mathrm{g}} \delta^{d_{\mathrm{g}}}$ and the quota in equation 109. Since $\delta^{d_{\mathrm{g}}} \leq 1$, the cost of all attempts in any episode satisfies

$$kc_e(\delta) \leq \left(\frac{\Lambda}{a_{\mathrm{g}} \delta^{d_{\mathrm{g}}}} + 1 \right) \frac{c_0}{\delta^{d_{\mathrm{v}}}} = \frac{c_0(\Lambda/a_{\mathrm{g}} + \delta^{d_{\mathrm{g}}})}{\delta^{\nu}} \leq \frac{c_0(\Lambda/a_{\mathrm{g}} + 1)}{\delta^{\nu}}. \tag{111}$$

Let $E_T \geq 1$ be the deterministic number of episode slots, including padding, and let $b_T^{\mathrm{gen}} > 0$ be the allocation for these attempts after reserving other costs. This is a resource allocation, distinct from the run cap b. Define

$$z_T = \frac{c_0 E_T (\Lambda/a_{\mathrm{g}} + 1)}{b_T^{\mathrm{gen}}}, \qquad 0 < z_T \leq 1, \qquad \delta_T = z_T^{1/\nu}. \tag{112}$$

The positive real root satisfies $0 < \delta_T \leq 1$ and $\delta_T^{\nu} = z_T$. Substituting into equation 111 and summing the E_T slots bounds total attempt cost by b_T^{gen}. Hence $b_T^{\mathrm{gen}} + B_T^{\mathrm{other}} \leq B_T$ ensures the original hard constraint. This is a sufficient, explicit design, not an infimum over incompatible parameter choices.

For $T \geq 2$, take $\Lambda = 2 \log T$. Then $e^{-\Lambda} = T^{-2}$, and the aggregate failure contribution is at most T^{-1} when $\sum_e n_e \leq T$. Thus equation 108 specializes to

$$\frac{R_T}{T} \leq \frac{\overline{U}_T + \sum_e \tau_e}{T} + \left[\frac{c_0 E_T (2 \log T/a_{\mathrm{g}} + 1)}{b_T^{\mathrm{gen}}} \right]^{1/\nu} + T^{-2}. \tag{113}$$

If the bracketed ratio tends to zero, the quality term tends to zero by continuity of a positive power at zero. Remaining terms must still vanish. For fixed a_{g}, c_0, the familiar sufficient scaling $E_T \log T = o(b_T^{\mathrm{gen}})$ makes that ratio vanish. When additional delay and deployment effects are bounded by a fixed constant times b_T^{gen}, one also chooses $b_T^{\mathrm{gen}} = o(T)$ and verifies the unaffected reference terms. Resource availability itself need not be sublinear, since an agent need not spend its entire allowance.

The values of coverage and cost envelopes determine this design. An algorithm adapting to unknown envelopes is a further learning problem, not a consequence of taking an infimum over designs. Any such optimization must retain the dependence of delay, observation, switching, and deployment terms on the same design.

J.4 Compatibility with the native verified-template instance

Corollary 12 and Corollary 13 use the same transfer theorem with different choices of the local generation parameters. In the verified-template instance, there is one episode with $k_e = k$, $n_e = N$, $\rho_e(0) = 1/2$, and $\delta_e = 0$. A trial costs two units, and registration plus the deployment check reserve

two further units. The hard bound is exactly $2k + 2$. Preparation is already charged as $F = k$, so $\tau_e = 0$ in the generation ledger. Its direct deployment charge remains $1/4$.

The generic exponential failure envelope becomes $e^{-k/2}$. The exact finite-model probability is 2^{-k} and is sharper. Keeping that exact bound preserves Corollary 12 verbatim. Its native controller and counterfactual comparators are unchanged. In particular, the new result does not demand the stronger global continuation-range contract discussed in Appendix I.

The Lean declaration `concrete_compatibility` checks both conclusions on the original model. It retains the actual prefix-cost function and derives the exponential-envelope regret bound from the already proved native regret. Thus compatibility includes the concrete execution and hard resources, rather than only a comparison of abstract rate expressions.

The polynomial specialization concerns $\delta > 0$ and illustrates a genuine precision-cost tradeoff. The existing exact-repair point has $\delta = 0$ and uses the more general corollary directly, rather than substituting zero into a negative-power cost formula. This distinction is the reason the two results are compatible. One establishes a nonempty concrete native model, while the other explains how a family of legal designs can allocate resources as quality requirements change.

Formal proof chain. Module `JointGenerationBudget` reuses the normalized finite trial law and proved failure recurrence. It constructs the missing generation certificate from episode kernels and exact endpoint identities, then calls `NativeFamily.own_coupling_counterfactual`. Direct or locally established deployment certificates remain explicit inputs. Its variable-domain family theorem invokes this native endpoint before proving the limit. Separate declarations verify every-prefix costs, integer quotas, polynomial costs, logarithmic failure targets, and positive-root choices. The scalar power manipulations are supporting steps rather than substitutes for the native semantic connection.

Interpretation of local generation reliability. Local generation reliability gives $\mathbb{P}_e(\xi_e = 1) \le (1 - \rho_e)^{k_e}$ by successive conditional failure bounds. Integrating task gaps and summing episodes proves equation 20. More trials reduce failure, better retained policies reduce quality error, and faster acquisition reduces delay. Theorem 11 combines these effects with both deployment bounds and the remaining contracts. The Lean family endpoint applies this completed-task bound before taking the no-regret limit. Appendix E supplies the proof.

All trials consume resources, including failures. Delay is charged here or in F, never both. Local generation reliability must hold uniformly over the comparator class. Increasingly precise, disjoint targets need not retain constant coverage. The remaining contracts must hold jointly.

K Notation, types, and correspondence to the formal model

The notation follows the usual distinction between losses ℓ, loss estimates $\widehat{\ell}$, unnormalized weights w, and sampling probabilities p in online learning (Lattimore & Szepesvári, 2020; Mourtada & Maillard, 2017). A bold $\mathbf{v}$ is a path, never its loss. The functions J_T and J with semantic superscripts denote expected totals, whereas L denotes a realized total. The separately defined J_r is a sampled reference identity, not a loss. Subscripts and superscripts distinguish the native model, reference slots, local stability model, and binary benchmark. Dummy variables in explicitly local proofs are not additional global model parameters.

K.1 Histories, probability laws, and conditioning

An operation history $h \in \mathcal{H}_s^{\mathrm{op}}$ records s completed internal operations. A task history $H_t \in \mathcal{H}_t^{\mathrm{task}}$ contains the initial record and t completed task records. These are not the same prefix space. A task-record kernel packages the internal computation between task boundaries, assuming it terminates and induces a measurable kernel. The theory does not prove termination of arbitrary language-model programs. The symbol E in Definition 1 is an admissibility relation, whereas $\mathbb{E}$ is expectation.

The runtime is the mechanism that interprets a selected card, invokes tools or primitive actions, and updates the recorded state. A card kernel $\pi_i(da \mid x, m)$ assigns a probability measure to actions a, conditional on context x and memory m. Here da is integration notation, not a product involving

an undefined parameter d. The same convention applies to the operation kernel $\mu(du \mid h)$. For a measurable function f, execution averages $f(a)$ as $\int f(a)\pi_i(da \mid x, m)$.

For a fixed comparator, Γ_μ is a joint proof law on a space Ω_μ. A realization is a measurable map from that space to a complete trajectory. The subscript # is the measure-pushforward operator, not a cardinality symbol. For a measurable trajectory event D, $((r_A)_\#\Gamma_\mu)(D) = \Gamma_\mu(r_A^{-1}(D))$, where $r_A^{-1}(D) = \{\omega \in \Omega_\mu \mid r_A(\omega) \in D\}$. Thus the probability of D is obtained by measuring the source outcomes mapped into it. The required equalities $(r_A)_\#\Gamma_\mu = \mathbb{P}_A$ and $(r_\mu)_\#\Gamma_\mu = \mathbb{P}_\mu$ preserve the complete native trajectory distributions, not merely one-step action marginals. The learner law $\mathbb{P}_A$ cannot change with the comparator used to prove the bound.

The notation $\mathbb{E}_{J_r, O_r \mid h_r}$ means a finite conditional expectation under masses $p_r(i)k_r(i, o)$ after fixing the completed pre-draw history. Equivalently, in a compatible reference process it is conditional expectation given $\mathcal{F}_r^-$. The sigma-field is generated by the completed prefix and contains no current draw or future observation. All slot-dependent objects may be functions of that prefix. The notation suppresses these arguments but does not assert determinism. By contrast, $\mathbb{E}_{\Gamma_\mu}$ integrates the entire coupled experiment, including its random prefixes. The law $\mathbb{P}^{(r)}_{\text{stop}}$ is a prefix marginal of the recursively generated stopped law. It is not conditioning on the terminal stopping time.

The endpoint $\ell_r^{\text{ref}}(i)$ is fixed before the current draw. Operationally it is the expected task-loss total obtained from the specified reference execution at that prefix and identity. The score $y_r(i, o)$ is an observable proxy. Equation equation 5 is a calibration assumption relating that proxy to the independently specified target. Defining the target to be an accepted-score average after the fact would not establish correspondence to task loss. Neither a positive $q_r(i)$ nor a positive sampling probability $p_r(i)$ supplies that correspondence.

Table 3: Semantic and algorithmic notation. Domains and probability laws are part of the definitions.

Symbol	Meaning and scope
T, t, s, r, R, N	Task horizon and task index, internal-operation index, reference-slot index, last padded slot, and slot cap $N = R + 1$. Defined in Sections 2 and 2.4.
$\mathsf{U}, \mathsf{O}, \mathsf{U}(h), \mathsf{E}$	Operation and result spaces, legal next operations, and admissible extension relation. These describe operational legality, not resource penalties.
$\mathsf{X}, \mathsf{M}, \mathsf{A}$	Context, memory, and primitive-action spaces. Lowercase x, m, a are their elements. The benchmark phase count m is local to Section 4.
$\mathcal{H}_s^{\text{op}}, \mathcal{H}_t^{\text{task}}$	Operation-history and task-history spaces. Random task prefix $H_t = (Z_0, \ldots, Z_t)$, with $Z_t \in \mathcal{Z}_t$. Lowercase h is a realized history.
$\pi_i, \mu, A, \mu_A, \mathcal{M}$	Execution-card action kernel, comparator meta-policy, complete native learner, its operation-selection kernel, and nonempty comparator class. i is immutable executable identity, not a primitive action.
$\sigma_t, Y_t, L_T, J_T, R_T$	Common task scorer, scalar task loss, realized cumulative loss, expected cumulative loss, and own-history regret from equation 1. Unqualified R_T abbreviates $R_T(A, \mathcal{M})$.
$\mathbb{P}_A, \mathbb{P}_\mu, \Gamma_\mu$	Own-trajectory laws and comparator-specific coupling with the prescribed native marginals. Ω_μ, ω denote its sample space and outcome.
$r_A, r_\mu, (r_A)_\#\Gamma_\mu$	Measurable trajectory realizations and the induced trajectory law. The subscript # denotes measure pushforward. The letter r alone indexes a reference slot.
Cost_t^μ, B_t	Cumulative resource use and deterministic hard budget at task boundary t. Non-task operationwise restrictions must be imposed separately.
$\mathcal{F}_r^-, h_r, \mathcal{I}_r$	Pre-draw sigma-field, realized completed prefix, and nonempty finite available identity set. The set and local kernels may be history-dependent.
$\mathbf{1}\{r \text{ active}\}$	Indicator that the reference process has not terminated and the completed history permits slot r. It is $\mathcal{F}_r^-$-measurable, not a card-availability or feedback-acceptance indicator.

Symbol	Meaning and scope, continued
ν_r, w_r, W_r	Probability prior, nonnegative unnormalized weight vector, and total weight $W_r = \sum_i w_r(i) > 0$. W_r is not a task record Z_t.
$p_r, J_r, \Theta_r, \alpha$	Sampling probability vector, sampled reference identity, row-stochastic sharing transition, and share rate in $(0, 1)$. The general transition need not use the Fixed-Share specialization.
k_r, O_r, M_r, q_r	Conditional outcome kernel, random outcome, binary acceptance mask, and acceptance probability conditional on the selected identity. q_r is not the probability of selecting that identity.
$\lambda_r, y_r, d_r, \ell_r^{\mathrm{ref}}$	Reference task-loss total, observed proxy score, its predictable upper bound, and reference expected loss per identity. Equation equation 5 links the two scores.
$\widehat{\ell}_r, e_r$	Importance-weighted estimated loss and zero-conditional-mean learner–comparator estimation noise. Squaring a vector means coordinatewise squaring.
$\mathbf{v}^\mu, v_r, C(\mathbf{v}), C$	Predictable available comparator path, its identity coordinate, realized negative log path mass, and deterministic uniform code bound. Dependence on μ is suppressed locally.
V_{est}, V	Raw cumulative second-moment budget in equation 11 and its scaled envelope with $V_{\mathrm{est}} \le bV$. Neither is a centered variance.
$\mathbb{P}_{\mathrm{stop}}, \varpi_r, F_r, f_r$	Stopped law, completed-prefix probability mass, cumulative event score, and its one-step increment in Appendix B. F_r there is not the scalar foreground budget F.
$\omega_r^{\mathbf{v}}$	Unnormalized contribution of a single supported path to weight $w_r(v_r)$, as defined in equation 47.

K.2 Endpoint types and the single accounting chain

Fix T and μ. Each intermediate endpoint is an explicitly chosen integrable score random variable on the coupling space. Its expectation is the displayed scalar J. The superscripts describe its role, not an algebraic operation that could manufacture the endpoint automatically. An application must supply the random variables and their semantic identifications before proving their inequalities. In particular, J_μ^{av} is an expected score, whereas $\mathbf{v}$ is a random predictable path.

Table 4: Expected endpoints, directed charges, and exact Lean field names. All expectations use Γ_μ.

Endpoint or charge	Definition, outgoing comparison, and correspondence
J_A^{nat}	Expected native learner loss $J_T(A)$. Expectation of `realA`. Its difference from J_A^{ref} is bounded by E_A.
J_A^{ref}	Expected reference learner endpoint, expectation of `refA`. Its difference from J^{sel} is bounded by $F + \mathcal{P}_b + D_A$.
J^{sel}	$\sum_{r=0}^{R} \mathbb{E}_{\Gamma_\mu} \ell_r^{\mathrm{ref}}(J_r)$, identified with `Experiment.selectedTotal`. Its comparison to J_μ^{av} is the proved selection bound.
J_μ^{av}	$\sum_{r=0}^{R} \mathbb{E}_{\Gamma_\mu} \ell_r^{\mathrm{ref}}(v_r)$, identified with `Experiment.availableTotal`. Its difference from J_μ^{al} is bounded by G.
J_μ^{al}	Aligned represented comparator score, expectation of `aligned`. Its difference from J_μ^{cl} is bounded by bS.
J_μ^{cl}	Score after removing alignment, expectation of `clean`. Its difference from J_μ^{path} is bounded by $h_{\mathrm{resp}}(S+1) + D_C$.
J_μ^{path}	Reference execution-path score after response comparison, expectation of `refPath`. Its difference from J_μ^{ref} is bounded by E_{nf}.

Endpoint or charge	Definition, outgoing comparison, and correspondence, continued
J_μ^{ref}	Reference comparator meta-policy score, expectation of `refM`. Its difference from J_μ^{nat} is bounded by E_C.
J_μ^{nat}	Expected native comparator loss $J_T(\mu)$, expectation of `realM`. No outgoing arrow is needed.
$E_A, E_C, E_{\mathrm{nf}}$	Learner deployment, comparator deployment, and representation budgets, corresponding to `Budget.E_cpA`, `E_cpC`, `E_nf`.
F, G, D_A, D_C	Foreground-displacement, generation, learner-calibration, and comparator-calibration budgets. The Lean `Budget` fields have these same names.
$b, h_{\mathrm{resp}}, N_b, S, V$	Nominal run cap, response-bound scale, extra-boundary envelope, path-switch envelope, and scaled capacity. Lean fields are `b`, `H`, `N`, `S`, `V`. In particular, `Budget.N` is N_b, not the slot cap N.
$\mathcal{P}_b, \Delta, U_T$	Prefix allowance, sum of seven scalar charges, and final upper bound, matching `Budget.prefix`, `delta`, and `bound`.
$g_T^{\mathrm{gen}}(\mu), G$	Exact signed endpoint difference $J_\mu^{\mathrm{av}} - J_\mu^{\mathrm{al}}$ and a uniform upper budget. Their relationship is $g_T^{\mathrm{gen}}(\mu) \le G$, not equality by definition.

The seven directed certificates in equation 13 are not seven independently optimized regret objectives. Together with the proved selection arrow they form the eight adjacent differences in equation 50. The scalar budgets may be signed when a tighter directed comparison permits it. The local generation theorem constructs a nonnegative envelope. No sign assumption is used to telescope the endpoints. Nonnegative multipliers are required when substituting upper budgets, such as $\eta/2$ for the capacity inequality.

K.3 Local-model and benchmark notation

Local generation reliability is the collective term for the success and task-quality conditions in Section 3.3. Success includes validation, retention, and causal usability. The parameters ρ_e and δ_e quantify these properties, while episode accounting and hard feasibility govern their composition. This term introduces no additional numeric parameter.

Table 5: Notation specific to stability, generation, and the paid binary benchmark.

Symbol	Meaning and scope
$S_t, s, K_t, L_t, p^{\mathrm{init}}$	Complete finite state, its realization, native and reference state-transition kernels, and common initial distribution in Section 3.2. The capital switch bound S is not a state.
$c_t, V_{t,n}^L, h_t, \bar{H}$	Common immediate state score, n-step continuation value from state-time t, local continuation-range envelope, and its uniform bound. $\bar{H}$ is distinct from the response scale h_{resp}.
$\mathrm{TV}, D_{\mathrm{KL}}, \varepsilon_t, \bar{\varepsilon}$	Total variation, relative entropy, local full-state TV envelope, and uniform activity-scaled TV factor. Relative entropy uses natural logarithms and the common support in Appendix F.
$\chi_t, r_t, c_{\min}, B_A, B_C$	Nonnegative update activity, actual-path resource charge, strictly positive cost per unit activity, and learner/comparator resource envelopes. Here $r_t(s)$ is a resource function, not a slot number.
$\zeta_t, \mathcal{D}_{t,n}, \kappa$	Local product $h_t\varepsilon_t$, its accumulated K-Bellman charge, and $\bar{H}\bar{\varepsilon}/c_{\min}$. The proof integrates along the native K law.
$e, k, k_e, \mathcal{H}_{e,k}^{\mathrm{gen}}$	Generation episode, trial index, number of trials, and episode-specific completed-trial space. Trials and task rounds need not coincide.
$\mathbb{P}_e, \varpi_k^{\mathrm{gen}}, g_k, b_k, \xi_k$	Episode law, prefix mass, generation kernel, success indicator in $\{0,1\}$, and all-failed indicator. Episode index e is suppressed inside the one-episode proof.

Symbol	Meaning and scope, continued
$\rho_e, \delta_e, \tau_e, n_e$	Conditional retained-card success lower bound, post-success quality tolerance, number of delay tasks, and number of later tasks in that episode.
$g_t^{\mathrm{pre}}, g_t^{\mathrm{post}}, \xi_e$	Conditional task-gap functions and failure indicator after k_e trials. The first two are not the trial kernel g_k. Their episode sums identify the actual generation endpoint.
$A_T, \delta_T, f_T, U_T^{\mathrm{rest}}$	Uniform delay, quality, and failure envelopes, and the reference bound with $G = E_A = E_C = 0$. A_T is a numeric delay allowance, not the learner A.
$\rho_e(\delta), c_e(\delta)$	Quality-dependent conditional coverage and prepaid per-attempt resource upper bound in Corollary 13.
$B_T^{\mathrm{other}}, C_A(h)$	Reserved other costs and total learner cost at legal prefix h.
$C^{\mathrm{other}}(h), u_e(h)$	Other charged costs and episode attempt count at legal prefix h.
$\overline{U}_T, \lambda_T$	Full transfer bound with only G removed, and a uniform lower bound on exposure $k_e\rho_e(\delta_e)$.
$F_{e,j}, \mathcal{F}_{e,j}$	No-success event after j trials and completed-trial information in episode e. The event $F_{e,j}$ is distinct from the foreground-loss budget F.
$\Lambda, \rho_\star, c_\star$	Log-failure target, coverage lower bound, and trial-cost envelope for the explicit quota in equation 109.
$a_{\mathrm{g}}, c_0, d_{\mathrm{g}}, d_{\mathrm{v}}, \nu$	Polynomial coverage coefficient, cost coefficient, nonnegative integer exponents, and their positive sum in equation 110.
$E_T, b_T^{\mathrm{gen}}, z_T$	Episode-slot count, attempt-resource allocation, and dimensionless ratio defining the positive quality root. None is the run cap b or a deployment-error certificate.
$m, n, B, j, K_j, j(t)$	Public phase count, phase length, total integer registration budget, phase index, phase library size, and phase containing task t. The benchmark uses $T = mn$.
$\mathcal{D}, \ell_t^{\mathrm{base}}, I_t, q$	Legal loss-table domain, fixed binary base loss, emitted identity, and primitive observation probability. The score masks are independent Bernoulli variables with parameter q.
$\mathcal{V}, \mathbf{v}, I_t^{\mathbf{v}}$	Deterministic eligible phase paths, one such path, and its emitted identity. Unlike the native comparator, this benchmark comparator is a phase path on an oblivious table.
$\mathfrak{A}, \mathcal{R}_{\mathrm{bin}}^*$	All causal randomized phase controllers and their minimax task regret in equation 27.
$\mathbb{P}_0, \mathbb{P}_j, \epsilon, N_j$	Null and hidden-target observation laws, Bernoulli mean gap, and the number of draws of identity j in the finite lower-bound proof.
$p_t^{\mathrm{low}}, x_j, y_j, E_q$	Conditional learner kernel in that proof, null and target expected draw counts, and the independent-erasure map on distributions.
$\mathcal{T}_r, u_r, X_i, V_{\mathrm{phase}}$	Task-index set of a held run, its length, its corrected loss under identity i, and phase-level second-moment envelope. Phase calculations suppress their phase index.

K.4 What the correspondence check does and does not establish

The paper notation is deliberately not forced to reproduce every internal Lean variable. In `DependentTransfer.Experiment`, fields `P`, `z`, `v`, and `V` denote Θ, $\widehat{\ell}$, $\mathbf{v}$, and V_{est}. Its measure parameter, locally named `μ` in mathematical presentations of the code, is the coupling law rather than a comparator meta-policy. The field `R` yields $R + 1$ scored slots. In `FiniteObservation.Round`, `kernel`, `accepted`, `score`, `loss`, and `q` correspond to k, M, y, ℓ^{ref}, and q. The target loss is an input field and the accepted-score identity is a premise. An operational scorer λ supplies a target instance, not a new theorem that every log is calibrated.

The native record convention follows `CausalTrajectory.History`, whose prefix includes coordinate zero. Task score t is evaluated after record $t + 1$, as in

`NativeSemantics.Execution.score`. The resource class in `ResourceClasses` has $T + 1$ task-boundary prefix coordinates. It does not silently quantify over an unrelated number of internal operations. Local generation indicators are ordinary $0/1$ values when multiplied, corresponding to Boolean tests with explicit coercion in the code.

The companion includes proved correspondence results without adding a numbered paper theorem. The family endpoint explicitly permits horizon-dependent finite domains and uses the combined-resource limit in Theorem 11. The source-and-log audit checks the included declarations and their dependencies. The interpretation of the English model and its operational identifications is manually reviewed, not mechanically proved by renaming notation. Integrability and exact score-law identifications remain part of the stated contracts.

The operational control kernel μ_A is not the literal type of `NativeSemantics.Execution.controller`. That Lean field is the induced task-record kernel after internal operations have been packaged together with their results and runtime effects. The interface assumes this packaging is well-defined. Renaming the operational kernel does not establish a new compiler-correctness theorem.

K.5 Terminology conventions

The *reference algorithm* denotes Algorithm 1, previously called a shadow algorithm. A *reference experiment* additionally specifies its probability law, scores, and comparator path. A *proxy score* is the observed scalar y_r, not a second algorithm. A *run* is a scored execution interval, a *slot* is its reference decision index, and a *generation episode* groups attempts and associated task gaps. These units need not have equal lengths.

The complete-execution objective R_T is called counterfactual regret to emphasize own-history policy-regret semantics. The signed generation opportunity gap $g_T^{\mathrm{gen}}(\mu)$ is one intermediate difference, and G is its upper bound. Neither is a newly introduced independent regret objective. A certificate is an explicit bound used by the proof, while Lean verification checks a conditional implication. An assumed certificate is not automatically an empirically validated fact. The paid binary benchmark is a restricted theoretical model, not a dataset or experiment.

Algorithm 1 exposes initialization, active and padded slots, and the final weight update of equation 3. The proof uses $N = R + 1$ scored slots and the outgoing update at $r = R$ to form W_{R+1}. That update creates no extra task or comparator switch. Supplied values of q_r and a lawful run protocol are interface inputs, not quantities automatically inferred from native logs. The pseudocode restates the analyzed recursion and does not claim a verified executable implementation of the native controller or generation scheduler.

K.6 Reading horizon-dependent conditions correctly

In a family of problems all uniform envelopes may depend on T. Uniformity means independent of the comparator at that fixed horizon, not constant as the horizon grows. To read Corollary 8, write these quantities as $b_T, h_{\mathrm{resp,T}}, N_{b,T}, S_T, C_T, V_T, \eta_T, \Delta_T$. For $C_T, V_T > 0$, choosing $\eta_T = \sqrt{2C_T/(b_T V_T)}$ makes the selection term $\sqrt{2C_T b_T V_T}$. Zero-capacity cases should use the original inequality, without dividing by zero.

For example, conditional on the required certificates, the scaling choices $h_{\mathrm{resp,T}} = O(1)$, $b_T = \lceil T^{1/3} \rceil$, $S_T = O(1)$, $N_{b,T} = o(T)$, $V_T = O(T)$, $C_T = O(\log(T + 1))$, and $\Delta_T = o(T)$ make each normalized term vanish. The selection term is $O(T^{2/3}\sqrt{\log(T + 1)})$. The nominal prefix term is $O(T^{2/3})$ and extra boundaries contribute $o(T)$. This illustrates the existing corollary, not an assertion that every growing agent supplies these envelopes. In particular, if births force $N_{b,T}$ to be linear, or the native/reference mismatch makes Δ_T linear, the example does not apply.

K.7 Symbols local to the concrete reflection instance

Appendix I uses k for the number of generic repair attempts, N for subsequent tasks and reference slots, and $T = k + N$ for the full task horizon. This scalar k is not an observation kernel $k_r(i, o)$. The programs P_0, P_1, P_2 are immutable execution identities, not probability laws. The trial bits ζ_j

Table 6: Roles of the results in the conditional native-transfer analysis.

Component	Role and scope of the contribution
Growing selection	Classical exponential weights and sharing, checked on history-dependent finite types.
Native transfer	Directed own-law composition, with application-dependent interface certificates.
Local stability	Range-sensitive TV telescoping, specialized to actual-path adaptive resource charging.
Local generation	Conditional coverage and per-task quality bound the generation opportunity gap.
Paid benchmark	Explicit common-model information lower and primitive upper, near matching only under equation 28.
Lean artifact	Reproducible verification of the conditional mathematical chain and its explicit interfaces.

form history H_k, and $\xi(h)$ is the all-failed indicator, consistent with the episode failure indicators used in the main generation result. The private bit D concerns the single deployment check. The procedures μ_0, μ_1 form the own-history class $\mathcal{M}_0$. The finite masses $w_k, s, Q_{\eta,h}$ define the coupling Γ in equation 99. The family sets $k = n + 1$, $N = k^2$, and $\eta = 1/k$. No new global regret definition is introduced.

L Clock alignment and validated mathematical interfaces

This appendix makes the interface boundary explicit. An interface is a premise about a specified model, not an unproved conclusion about its regret. The additional proofs below are checked in `CompletedClock`, `CompletedNativeV19`, and `ObservationBridgeV19`. They do not introduce an external controller.

L.1 Completed tasks and state-score positions

The native score $Y_t = \sigma_t(H_{t+1})$ is produced after task t. Take the finite sufficient state S_{t+1} to contain the completed history, and write $g_t(S_{t+1}) = Y_t$. In this appendix g_t is the completed-task scorer, not a generation gap. Define the padded state scorer by

$$\widetilde{c}_0(S_0) = 0, \qquad \widetilde{c}_{t+1}(S_{t+1}) = g_t(S_{t+1}). \tag{114}$$

There are T tasks and $T + 1$ scored state positions. A terminal dummy transition after S_T has zero resource cost and no effect on a scored task. The two processes share a pre-task initial state law, not the law of their first completed score.

For a kernel family K, let $W^K_{t,n}(s)$ be the expected score of the next n completed tasks. Its recursion and its relation to the state-score value are

$$W^K_{t,0}(s) = 0, \qquad W^K_{t,n+1}(s) = \sum_j K_t(s,j)\big[g_t(j) + W^K_{t+1,n}(j)\big], \tag{115}$$

$$V^{K,\widetilde{c}}_{t+1,n+1}(s) = g_t(s) + W^K_{t+1,n}(s). \tag{116}$$

For $n = 0$, the second identity says that one state position contributes its current score. For the induction step, expand one state transition and apply the same identity at each destination. Nonnegative normalized kernel weights preserve the equality. Expanding the finite trajectory law also proves that W equals the expectation of the sum of completed-task scores. Consequently

$$J^{\mathrm{task}}_T(K) = \mathbb{E}_{\mathbb{P}_K} \sum_{t<T} g_t(S_{t+1}) = \mathbb{E}_{\mathbb{P}_K} \sum_{s=0}^{T} \widetilde{c}_s(S_s) = J^{\mathrm{state}}_{T+1}(K, \widetilde{c}). \tag{117}$$

The last dummy transition integrates to one. Thus neither the first nor the last task score is removed. The Lean declarations are `post_trace_value`, `shifted_value`, and `completed_clock`.

Apply the state-score stability theorem at length $T + 1$ using $\widetilde{c}$. Choose the terminal update identically on both sides, with zero local charge and resource. The actual task budget, generation task count,

reference bound U_T, and regret normalization remain indexed by T. The resulting completed-task inequality is

$$|J_T^{\text{task}}(K) - J_T^{\text{task}}(L)| \le \bar{H}\bar{\varepsilon}B_T/c_{\min}. \tag{118}$$

This is `CompletedClock.completed_local_gap`. With no influence improvement the safe padded local-only bound is $(T+1)^2\bar{\varepsilon}_T$. Its normalized sufficient condition is still $T\bar{\varepsilon}_T \to 0$ as $T \to \infty$.

L.2 A validated instance and the exact composition

The publication-level `CompletedNativeV19.Instance` requires unit-range completed-task scores, positive task and run counts, nonnegative resources and activities, zero terminal resources, and integrability of every intermediate expected-score endpoint. The local state contracts use $T+1$ positions. The reference budget uses T actual tasks. No equality between the two clocks is silently imposed.

The remaining reference, episode, and own-law identities are still explicit. For each fixed comparator, the generation theorem gives its budget $G = A_T + T(\delta_T + f_T)$. Instantiate the reference certificates with this budget and zero inner deployment arrows. Convert the two reference identities using equation 117, invoke the local transfer at the padded state horizon, and convert the two native endpoints back to completed tasks. The reference task bound is unchanged. Thus

$$R_T \le U_T^{\text{rest}} + A_T + T(\delta_T + f_T) + \kappa(B_A + B_C), \qquad \kappa = \bar{H}\bar{\varepsilon}/c_{\min}. \tag{119}$$

The checked endpoint is `CompletedNativeV19.Instance.bound`. Its variable-domain family theorem invokes this bound before taking limits, using the actual task denominator T. It replaces the unpadded family endpoint as the publication correspondence for Theorem 11. Horizon-dependent model families need not be consistent prefixes of one anytime implementation. No such stronger implementation claim is made.

L.3 Connecting local observations to the same experiment law

Let h range over finite pre-draw histories, with probabilities $a(h)$. At each h, a finite observation round has the paper's accepted-score contract, draw probability p_h, and outcome kernel k_h. A law witness maps a complete sample ω to its history, draw, and outcome. For every finite test function f it requires

$$\mathbb{E}_{\Gamma_\mu} f(h, J, O) = \sum_h a(h) \sum_i p_h(i) \sum_o k_h(i, o) f(h, i, o). \tag{120}$$

This specifies the joint law, not merely an identity marginal. Histories can have different finite identity types. Inactive slots can use a singleton absorbing event. For use with the selection recurrence, p_h must be its actual pre-draw probability and the estimator must be its actual update. The comparator coordinate is a function of h, not of the current outcome.

Apply the already proved local first-moment identity separately at each h, multiply by $a(h)$, and sum. Apply the local second-moment inequality in the same way, using $a(h) \ge 0$. This gives

$$\mathbb{E}_{\Gamma_\mu}\left[\langle p, \widehat{\ell}\rangle - \widehat{\ell}(v)\right] = \mathbb{E}_{\Gamma_\mu}[\ell^{\text{ref}}(J) - \ell^{\text{ref}}(v)], \tag{121}$$

$$\mathbb{E}_{\Gamma_\mu}\langle p, \widehat{\ell}^2\rangle \le \sum_h a(h) d_h^2 \sum_i q_h(i)^{-1}. \tag{122}$$

`LawWitness.moment`, `LawWitness.second`, and `identified_moment` verify these implications and the pointwise pullback identification. The law witness and recurrence identification must be instantiated for a concrete model. There is no claimed automatic constructor from arbitrary native logs to a growing reference execution. The general reference experiment continues to expose these law and moment interfaces. This is the precise boundary of the conditional verification claim.

L.4 Accounting is determined by endpoints

For three endpoints, $a - c = (a - b) + (b - c)$, checked by `adjacent_not_causal`. For example, native, reference, and ideal expected losses $T, T/2, 0$ require both adjacent charges $T/2$. Both can be consequences of generation failures without being duplicate charges. Only charging the same endpoint

difference twice is prohibited. Conversely, an outer full-state charge already includes propagation of its own perturbation, so those same consequences must not be added again to that same arrow. A physical resource may constrain feasibility and affect a task-score difference, but feasibility alone does not add another loss penalty.

L.5 Joint feasibility and the scope of sublinearity

From $\mathrm{TV}(K_t, L_t) \le \bar{\varepsilon}\chi_t$ and $c_{\min}\chi_t \le r_t$, finite summation gives

$$\sum_{t<T} \mathrm{TV}(K_t(S_t,\cdot), L_t(S_t,\cdot)) \le \bar{\varepsilon} B_T / c_{\min}. \tag{123}$$

This is `accumulated_tv`. Also, zero resource implies zero activity and zero local TV, as proved by `no_resource_no_error`. Thus charging only rare reflection calls cannot cover persistent differences in ordinary selection. Constant-scale discrepancies on a positive fraction of rounds are incompatible with a sublinear resource envelope at fixed $\bar{\varepsilon}, c_{\min}$.

Coverage attempts, delays and hard budgets must be feasible in the same instance. Corollary 12 supplies a concrete native-transfer witness using direct deployment certificates. The stronger full-state range contracts of the local-resource variant still require their own justification. Corollary 13 derives feasible quotas under quality-dependent coverage and cost contracts. Neither result turns separately desirable exponents into a universal model guarantee. Trial resource use must follow from the same legal design.

For the separate sparse benchmark, divide $q^3 T \le mB \log B$ by the positive qT to obtain

$$q^2 \le \frac{mB \log B}{qT}. \tag{124}$$

The finite inequality is `sparse_normalized`. It shows why the sparse-family sufficient rate condition requires vanishing observation probabilities. It is not an impossibility result for fixed positive observation probability. The public exogenous registration schedule also does not establish necessity of every endogenous native certificate.

M Related work and research connections

Availability and generation. Sleeping bandits address availability (Saha et al., 2020; Nguyen & Mehta, 2024), ballooning models address arriving arms (Ghalme et al., 2020), and growing-expert tracking aggregates expanding pools (Mourtada & Maillard, 2017). Generative action sets use a context-conditioned oracle, permanent reuse, quadratic mismatch and a doubly-optimistic creation-selection controller (Xu et al., 2026). Our Theorem 11 connects causally usable policies to complete native executions through delay, quality and conditional failure. It permits dependent trials. Strongly adaptive regret offers an interval-based perspective (Daniely et al., 2015), while our episodes partition the available-to-aligned endpoint difference.

History, feedback, and switching. Classical bandit theory connects specified controllers and observation laws to regret (Lattimore & Szepesvári, 2020; Bubeck & Cesa-Bianchi, 2012). Policy regret (Arora et al., 2012), memory-dependent losses (Anava et al., 2015; Cesa-Bianchi et al., 2013), and Markov games (Nguyen-Tang & Arora, 2025) ground history-responsive comparisons. Our transfer retains the native operation-selection kernel. Theorem 9 bounds full-state discrepancies along actual resource paths. Theorem 11 combines this charge with local generation bounds through ordered score endpoints.

Switching-cost bandits (Dekel et al., 2014; Rouyer et al., 2021), feedback-sensitive bounds (Cheng et al., 2023), and feedback graphs (Alon et al., 2015) characterize the role of observation and action-change protocols. Our reference analysis specifies accepted-score calibration and proves the associated observation moments. The supporting model of Theorem 14 uses independent primitive erasures and paid public growth, giving near-matching bounds in the stated sparse regime with common learner, comparator, and cost conventions.

Mechanized learning guarantees. Merten et al. (2018) verified multiplicative weights and composed distributed-system guarantees in Coq. Our Lean artifact connects native execution, growing selection, observation, generation, deployment and completed-task clocks. It verifies concrete native instances and a joint quality-budget design. Declaration mappings identify the contracts and endpoints of all thirteen numbered results.

Rule-card identities select action kernels, connecting exponential weighting to execution semantics. Full-state stability and conditional failure bounds connect local reflection to total loss. Table 6 summarizes these component roles.

Connection to native-agent evaluation. Empirical studies of native LLM online play (Park et al., 2025) motivate analysis of the controller used by the agent. Theorems 9 and 11 connect local generation reliability and full-state deployment stability to the complete native objective. Their joint control determines when the conditional bound is informative.

Off-policy evaluation (Dudík et al., 2011) supports analysis of logged decisions. Our reference process additionally requires an identified history law, observation support and accepted-score calibration. Appendix G connects these operational requirements to the transfer interfaces.

N Repeated growth with imperfect validation

This second native instance makes validation and subsequent selection genuinely fallible. New context-specific policies are generated in several phases. A false acceptance can overwrite a previously correct native choice and cause errors throughout the remaining phase. The controller follows a recency rule, never runs the reference Bandit, and has no deterministic repair prompt. The construction leaves Theorems 7 and 11 unchanged. It provides a sharper direct calculation and separately verifies compatibility with the existing transfer interfaces.

N.1 A frozen generator and an imperfect executable validator

Fix integers $E, k, m, N \geq 1$ and $T = E(k+N)$. There are E public task types, presented in consecutive phases. In phase e, each deployment task has input $x = 1$ and requests output one. Candidate programs are immutable and type-tagged. A correct type-e program returns x on type e and zero on other types. An incorrect candidate returns zero. The initial fallback also returns zero. Correct programs for different types induce different action kernels. Previously registered cards remain in memory, but the legal execution interface calls only current-type cards or the fallback.

The same frozen generation kernel, queried with the current type, independently produces a correct or incorrect candidate with probability $1/2$. No quality label is returned to the controller. A generated handle names immutable content and its version. Each attempt has a fresh handle, even when its behavior duplicates another candidate. Semantic growth across types is distinguished below from the raw number of registrations.

The validator tests a candidate on m independent fair inputs in $\{0, 1\}$ and accepts only when all outputs match their inputs. A correct current-type program always passes. An incorrect program passes only when every probe is zero. Thus its conditional false-acceptance probability is

$$\beta_m = 2^{-m} > 0. \tag{125}$$

This is an imperfect finite-test validator, not a complete program verifier. The sampling protocol is fixed. The native procedure receives candidate handles, test inputs and outputs, and the acceptance decision. It never receives the latent correct/incorrect label. All procedures use the same generation and validation operations. The environment in this instance is deterministic apart from the specified generation and validation randomness.

For analysis, a completed attempt has one of three outcomes, denoted $0, 1, 2$, with probabilities

$$w_m(0) = \tfrac{1}{2}, \qquad w_m(1) = \tfrac{\beta_m}{2}, \qquad w_m(2) = \tfrac{1-\beta_m}{2}. \tag{126}$$

Outcome zero is a correct accepted card, one an incorrect accepted card, and two a rejection. These probabilities follow by pushing the fair generation bit and the actual m probe bits through the validator. Outcomes zero and one have the same acceptance signal. The controller can therefore use the acceptance bit without observing this three-way analysis label.

N.2 Native execution, retention, and hard resources

In each phase, the native learner executes the fallback for the first k tasks and performs one generation attempt after each task. Every accepted candidate is immediately registered and retained. Each acceptance updates a current-type pointer to that newest handle. Rejections preserve the pointer. The following N tasks execute the pointer, or the fallback if none was accepted. The pointer is not repaired during this suffix, even when a task reveals an error. This is an explicit native batching rule, not a claim about every reflective system.

Registration does not force immediate deployment. The batch choice pays for its k preparation tasks. The interface still permits other native schedules, including immediate use after a completed registration and a new predictable draw. The construction does not impose a block-boundary restriction on the general model.

Generation costs one unit, each probe costs one unit, and each accepted registration costs one unit. Before initiating an attempt, its worst-case cost $m + 2$ is reserved. Generation and probe costs of failed attempts remain charged. Any unused registration reserve is released only after the attempt completes. Every prefix, including an unfinished attempt, then satisfies

$$C_A(h) \le \sum_{e<E} (m+2)u_e(h) \le Ek(m+2) = B_T, \qquad 0 \le u_e(h) \le k. \tag{127}$$

Here $u_e(h)$ is the number of initiated attempts in phase e. No registration is possible before initiation. The common prefix cap is B_T, available from the start. Stronger time-varying resource schedules would require their own feasibility checks. Resources constrain legal execution and are not added a second time to the task loss.

The comparator class $\mathcal{M}$ is any nonempty class of complete legal hard-feasible procedures with the same task scorer. For the separation from no reflection, require that it contains the following proactive procedure. At the start of each announced phase, it completes the same k random generation and validation attempts, uses the same latest-accepted rule, and then executes all $k + N$ tasks. It pays the same worst-case budget and has neither a stronger generator nor a perfect validator. Its observations and memory are generated on its own execution, not copied from the learner.

Proposition 16 (Repeated native growth with imperfect validation)**.** *The specified native procedure is hard-feasible under equation 127. Let $q_{m,k}$ be the probability that its phase pointer is incorrect after k attempts, including the fallback. Then*

$$q_{m,0} = 1, \qquad q_{m,j+1} = \tfrac{\beta_m}{2} + \tfrac{1-\beta_m}{2} q_{m,j}, \qquad 2^{-k} \le q_{m,k} \le 2^{-k} + \beta_m. \tag{128}$$

The complete own-law objective satisfies

$$J_T(A) = E[k + Nq_{m,k}], \qquad R_T(A, \mathcal{M}) \le E[k + N(2^{-k} + 2^{-m})]. \tag{129}$$

The expected number of task types for which a genuinely correct retained policy has been generated is $E(1 - 2^{-k})$. If $k, m \ge 2$ and the proactive comparator belongs to $\mathcal{M}$, the no-reflection procedure has regret at least $T/2$. With $E = k = m \to \infty$ and $N = k^3$, both positive normalized native regret and normalized hard resources tend to zero.

N.3 Proof from the actual execution law

Let H_j be the completed outcome history in one phase. Its law is recursively generated from equation 126. Let $Z_j(H_j) \in \{0, 1\}$ indicate whether the selected pointer is incorrect. Initially $Z_0 = 1$. An outcome-zero acceptance sets it to zero, an outcome-one acceptance sets it to one, and a rejection retains the old value. Importantly, the control decision in both acceptance cases is identical, namely point to the newest handle. Only analysis of the program output distinguishes the cases.

To connect this recursion to executable program content, give attempt j the local handle j and pair it with its phase tag. The registry stores accepted handles without deleting older entries. Induction on j proves that every nonempty pointer names an accepted handle with index less than j. A second induction proves that the correctness of the program at that pointer is exactly $1 - Z_j$. Rejection preserves both the old pointer and the immutable content at its index. Acceptance points to the new content, irrespective of its latent quality. Thus Z_j is the actual program loss at deployment input one,

not an independently assigned surrogate score. The same registry induction proves that a correct retained program exists exactly when at least one outcome-zero attempt has occurred.

Conditioning on the next outcome O_{j+1} after the completed prefix gives

$$\mathbb{E}_{O_{j+1}|H_j}[Z_{j+1}] = \tfrac{\beta_m}{2} + \tfrac{1-\beta_m}{2} Z_j. \tag{130}$$

Integrating this equality yields the recursion in equation 128. This computation derives $q_{m,j}$ from the native pointer update rather than assuming a small selection error. For the upper bound, suppose $q_{m,j} \le 2^{-j} + \beta_m$. Since $0 \le \beta_m \le 1$,

$$q_{m,j+1} \le \tfrac{\beta_m}{2} + \tfrac{1-\beta_m}{2}(2^{-j} + \beta_m) \tag{131}$$

$$= 2^{-(j+1)} + \beta_m - \beta_m 2^{-(j+1)} - \tfrac{\beta_m^2}{2} \le 2^{-(j+1)} + \beta_m. \tag{132}$$

The initial inequality holds because $q_{m,0} = 1 \le 1 + \beta_m$. If no correct candidate was generated, the pointer must be incorrect. This event has probability 2^{-k}, proving the lower bound in equation 128.

For every realized phase history, its actual sum of task scores is $k + NZ_k(H_k)$. Summing over the recursively generated law gives $k + Nq_{m,k}$. The full execution retains old libraries but each current-type kernel ignores their contents, so composing the phase laws gives the first equality in equation 129. Equivalently, linearity of expectation applied to the actual phase rewards gives the same identity. Since every complete comparator has nonnegative common task loss, its own-law expected loss is nonnegative. Subtracting the nonnegative infimum proves the regret upper bound. This argument applies to the original legal comparator class, rather than granting the comparator an unavailable card or a zero-loss oracle.

Every generated correct card is accepted and retained. Consequently a type has a correct retained policy exactly when at least one of its k generation bits is correct. The expected number of such types is $E(1 - 2^{-k})$. This counts semantically different type-specific policies, not duplicate handles within one type. The expected total number of accepted registrations, including duplicates and incorrect cards, is separately $Ek(1 + \beta_m)/2$.

The proactive comparator uses the same random pointer distribution but performs preparation before its tasks. Its own-law objective is

$$J_T(\mu_{\text{pro}}) = E(k + N)q_{m,k} = Tq_{m,k}. \tag{133}$$

For $k, m \ge 2$, the upper bound on $q_{m,k}$ is at most $1/2$. The no-reflection procedure has loss T and therefore regret at least $T - J_T(\mu_{\text{pro}}) \ge T/2$. This comparator is another noisy reflective procedure with its own generation history, not an omniscient policy. All resource prefixes of both procedures obey the same cap.

N.4 Why the validation error is not an instantaneous charge

Let M_k indicate that no correct card was generated. Define a clean analysis execution that, after the same preparation history, uses a retained correct card if one exists. Then $M_k \le Z_k$ for every history, and

$$\mathbb{E}_{H_k}[M_k] = 2^{-k}, \qquad 0 \le \mathbb{E}_{H_k}[Z_k - M_k] \le \beta_m. \tag{134}$$

The clean execution is not the deployed controller and is not assumed to run a perfect validator. It is an intermediate comparison object. Its advantage over the native pointer is proved from the actual three-outcome law and charged explicitly.

For example, a correct acceptance followed by an incorrect acceptance has probability $\beta_m/4$. The correct card remains registered, but the native pointer now calls the incorrect one. In fact, for two attempts the excess error is exactly

$$q_{m,2} - 2^{-2} = \beta_m/4 > 0. \tag{135}$$

Thus an available correct policy does not make the native selection bridge free. A wrong pointer remains active throughout N suffix tasks, so the deployment edge is at most $N\beta_m$ per phase. Counting only β_m would miss this continuation effect. The growing family below compensates by improving validation as the suffix grows. It does not assume that the global continuation-value range is one.

N.5 Connection to the unchanged transfer theorem

Condition on a completed phase history before constructing any reference draw. Use the same two-arm reference recursion as Appendix I, with losses one and M_k. The second arm represents an actually retained correct card when one exists, and an incorrect fallback otherwise. The clean native suffix has loss NM_k, which is no larger than the reference expected selection loss. The reference initialization, feedback law, code $\log 2$, and second-moment budget $2N$ are the previously proved ones. The finite controller form of Theorem 7 therefore has the following phase certificates

$$F = k, \qquad G = N2^{-k}, \qquad E_A \le N\beta_m, \qquad E_C = D_A = D_C = E_{\mathrm{nf}} = 0, \tag{136}$$

with response and switch charges zero inside this phase comparison. The all-correct ideal path has value zero, a valid lower comparison endpoint for any nonnegative own-law comparator value. It is not claimed to be a feasible zero-loss comparator. The direct calculation avoids paying the additional reference selection bound, whereas transfer gives the valid, generally looser inequality

$$R_T(A, \mathcal{M}) \le E\left[k + N(2^{-k} + 2^{-m}) + \frac{\log 2}{\eta} + \eta N\right], \qquad \eta > 0. \tag{137}$$

Each phase uses an independently constructed reference analysis after its preparation has completed. These restarts do not reset the native library. We do not silently invoke a single two-arm fixed-library process across all growing phases.

More explicitly, if $w_{m,k}(h)$ is the phase history law and $Q_{\eta,h}(g)$ the actual reference recursion law, define $\Gamma(h, g) = w_{m,k}(h)Q_{\eta,h}(g)$. Summing over g returns $w_{m,k}(h)$, and the native task score does not depend on g. Thus the reference extension preserves the complete native phase marginal. Its phasewise composition preserves the full native execution used above. Comparator objectives retain their respective own laws. This is a causal auxiliary construction, not offline reconstruction from unsupported action logs.

The three costs in equation 136 have different endpoints. Preparation losses occur before the suffix. Generation loss is the suffix failure event $M_k = 1$. Additional deployment loss is the event $Z_k = 1, M_k = 0$. These last two events are disjoint. Failed validation and candidate registration consume resources in equation 127, not an additional loss penalty. This prevents charging the same false acceptance as both generation failure and deployment discrepancy.

N.6 An explicit family with growing libraries and sublinear resources

Take $k \ge 2$, $m = E = k$ and $N = k^3$. Then

$$T = k^4 + k^2, \qquad B_T = k^2(k+2), \qquad \frac{[R_T(A, \mathcal{M})]_+}{T} \le k^{-2} + 2^{1-k}, \qquad \frac{B_T}{T} \le k^{-1} + 2k^{-2}. \tag{138}$$

Both right sides tend to zero. The first follows by dividing equation 129 by $T \ge k^4$. The resource calculation uses the same denominator, not a different budget design. The expected number of distinct repaired task types is $k(1 - 2^{-k})$, which grows without bound. For the transfer route, choosing $\eta = 1/k$ adds at most $(\log 2)k^{-2} + k^{-1}$ to the normalized direct bound, also tending to zero.

The guarantee is a horizon-indexed family, not an anytime claim. The restrictions are explicit, including independent template sampling, one-sided validation noise, public type-local phases and no cross-type effect on future generation. The instance supplies repeated semantic growth and persistent imperfect-validation errors within these restrictions. It does not establish joint tightness of the general upper bound or automatic guarantees for arbitrary language-model validators.

N.7 Formal correspondence

Module `ImperfectGrowth` connects the native registry and program loss to the probability model through `actual_execution`. Its `actual_regret` and `actual_no_regret` prove the bound and limit, allowing horizon-dependent comparator classes. The `full_transfer` proof invokes the unchanged transfer through the phase certificates. Appendix H and the declaration mapping cover the intermediate validation, retention and resource proofs. This verifies the specified mathematical program, not deployed language-model code. Reference learning is not necessary for the stronger direct calculation.