DWELL: PERSIST OR WRAP UP? MEASURING AND GOVERNING DYNAMICAL INSTABILITY IN LLM MULTI-AGENT ORCHESTRATION

Anonymous authors
Paper under double-blind review

ABSTRACT

When tools repeatedly fail, should a large language model (LLM) orchestrator persist or wrap up? Stopping early can lose recoverable successes, while continued retries can exhaust the budget on unsolvable tasks. We study this decision by separating observed progress from the allocation of further retries. We introduce a trajectory-level stability margin to measure progress and a budget-aware dwell-time controller, DWELL, that determines how long to keep retrying from the failure history and remaining budget under a simple recovery model. Experiments show that task success declines gradually as tool failures become more persistent and drops to zero when the last source of required information is lost. Comparisons with fixed stopping rules quantify how earlier stopping trades recoverable successes for resource savings. Within this trade-off, exploratory analysis on the primary frozen faulted workload finds that DWELL reduces tokens per successful task by 27.5% relative to ungoverned orchestration. Tests on new tasks with equal approach show that models stop when instructed and succeed when tools recover before the retry limit. Whether stopping helps depends on the task and how long the model would otherwise keep trying.

1 INTRODUCTION

A tool fails, and an LLM orchestrator retries. After several identical failures, should it persist or wrap up? This decision arises in workflows that interleave reasoning and tool use under execution failures and noisy observations (Yao et al., 2023; Froger et al., 2025). A temporarily rate-limited tool may recover on the next call; a permanently unavailable source may consume the remaining budget. These cases can produce the same observed failure prefix yet call for opposite decisions, and parallel dependencies complicate matters further: one executor can fail while other branches advance. Reliable orchestration requires both measuring overall progress and deciding whether continued effort is worthwhile.

Interactive benchmarks and process supervision expose intermediate progress (Liu et al., 2024; Ma et al., 2024; Lightman et al., 2024), failure taxonomies identify coordination and termination problems (Cemri et al., 2025), reliability surfaces parameterize repetition, perturbation, and injected faults (Gupta, 2026), and self-healing orchestrators and BENCH2ROBUST study budgeted recovery actions under controlled fault scenarios (Babu & Agrawal, 2026; Chen et al., 2026). These studies connect failures to recovery strategies, but a warning of lost progress cannot by itself establish whether another retry is valuable: a stall can remain recoverable, and the same intervention can help an early-conceding model but interrupt one that would have recovered through persistence.

Drawing on the within-mode contraction versus switching distinction in switched-system analysis (Hespanha & Morse, 1999; Liberzon, 2003), we formulate orchestration as a stochastic hybrid trace: executor identities define discrete modes, and unresolved work, failures, repetition, and context load define a task potential. A trajectory stability margin, $\mathcal{M}$, measures contraction along the realized executor sequence, a sliding-window form supplies an online stall warning, and an in-band observer estimates progress from the user query and executor traffic, keeping privileged benchmark checkpoints confined to evaluation. The warning has a scoped interpretation: the observed trajectory has lost

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.

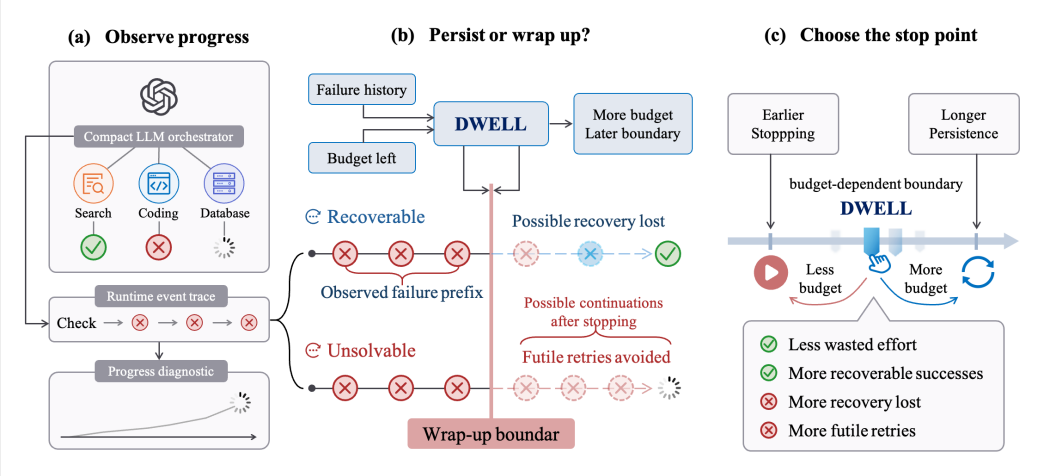


Figure 1: DWELL: Progress monitoring and retry governance.

effective progress. Selecting an intervention additionally requires assessing recovery opportunities and the remaining budget.

Decision-theoretic metareasoning relates continued computation to its expected benefit and cost (Russell & Wefald, 1991; Hay et al., 2012), restart theory budgets independent randomized executions (Luby et al., 1993), and recent agent studies examine feasibility-based stopping and calibrated early abort (Lin et al., 2026; Ruan et al., 2026). We instantiate a complementary approach in DWELL a stateful dwell-time governor that combines a posterior recoverability model with the remaining budget: the observed failure streak determines a posterior predictive chance of success within the remaining calls, a retry regime encourages persistence while recovery remains sufficiently plausible, and a wrap-up regime limits continued expenditure. DWELL acts through the existing advisory channel. Figure 1 summarizes the framework.

On the persistent deployments studied, stopping rules save resources on unsolvable tasks while sacrificing some recoverable successes. DWELL occupies a point on a patience-stop-loss frontier, with practical value conditional on the workload and the cost of premature stopping. These conclusions are scoped to workflows whose progress can be represented explicitly and whose failures appear mainly as persistent retries.

2 A Measurable Stability Margin

2.1 Orchestration as a Stochastic Hybrid Trace

Consider an LLM orchestrator that coordinates multiple executors to complete a task through iterative planning and tool interaction. A run with step budget B consists of a sequence of executor interactions $((\sigma_1, u_1, y_1), \ldots, (\sigma_T, u_T, y_T))$, $T \leq B$, where σ_k is the selected executor (the discrete mode), u_k is its request, and y_k is its output. At step k, the runtime system can use only the user query and executor traffic observed up to that point. We denote this available information by $\mathcal{F}_k$.

For analysis, we define a set of task-specific gold checkpoints, i.e., predefined evaluation milestones that represent distinct pieces of task progress. Different percentages of gold checkpoints completed represent different levels of task completion. We therefore define progress $z_k \in [0, 1]$ at step k as the fraction of all gold checkpoints supported by evidence observed up to that step. Further more, to capture the unresolved progress, we define

$$V_k = w_u(1 - z_k) + w_f F_k + w_c C_k + w_\ell R_k, \tag{1}$$

where $1 - z_k$ measures unresolved work, F_k and R_k are exponential moving averages of failed and repeated calls, and C_k is capped cumulative context load. The weights (w_u, w_f, w_c, w_ℓ) are pre-specified constants (Appendix A.2) that control the relative contribution of these components, so larger V_k indicates more unresolved work or accumulated execution burden.

2.2 Within-Mode Contraction and Switching Jumps

Because a smaller task potential V_k indicates less unresolved work and execution burden, we measure the change across each valid transition by the log-potential ratio $g_k = \log\big((V_{k+1} \vee \epsilon)/(V_k \vee \epsilon)\big)$, where $\epsilon > 0$ is a small constant. Thus, $g_k < 0$ indicates contraction of the task potential, reflecting progress toward task completion, whereas $g_k > 0$ indicates deterioration of the state.

To distinguish progress within an executor from the effect of switching executors, we separate transitions according to whether the active mode changes:

$$\overline{\log \rho} = \mathrm{mean}\{g_k : \sigma_k = \sigma_{k-1}\}, \qquad \overline{\log \mu} = \mathrm{mean}\{g_k : \sigma_k \neq \sigma_{k-1}\}, \qquad \nu = \frac{N_{\mathrm{switch}}}{T-1}. \quad (2)$$

Here, N_{switch} is the number of transitions for which $\sigma_k \neq \sigma_{k-1}$, and ν is the corresponding fraction of executor-switch transitions among the $T-1$ consecutive transitions.

Combine the within-mode and switching contributions into the trajectory-level stability margin

$$\mathcal{M} = -\left[(1-\nu)\overline{\log \rho} + \nu\overline{\log \mu}\right]. \quad (3)$$

While g_k measures the local change in task potential over a single transition, $\mathcal{M}$ aggregates these changes over the trajectory while accounting for executor switching. A positive $\mathcal{M}$ indicates that the task potential decreases on average, whereas a negative $\mathcal{M}$ indicates that unresolved work or execution burden grows overall. Thus, $\mathcal{M}$ provides a trajectory-level signal of whether the orchestration process is making sustained progress. For completed trajectories we also report the average-dwell diagnostic $M^{\mathrm{doc}} = -(\overline{\log \rho} + \nu\overline{\log \mu})$, used for the condition means and AUCs in Sections 3–4; online alarms use the windowed $\mathcal{M}_k$ below.

During task execution, we compute a runtime margin $\mathcal{M}_k$ from the most recent 6 steps using the same construction with $\mathcal{M}$. The estimate becomes available once at least three valid transitions have been observed. An alarm is raised when $\mathcal{M}_k < \tau$, indicating that the recent trajectory has lost effective contraction and may be entering a stall. The alarm serves as a runtime warning rather than a stopping decision; Section 5 determines how the system responds to such warnings. We use the frozen threshold $\tau = -0.02$ detailed in Appendix A.2 for offline detection and false-alarm rates; first-alarm timing (Section 3.4) and the deployed Governor use $\tau_{\mathrm{on}} = 0$, the latter with release threshold $\tau_{\mathrm{off}} = 0.05$ (Appendix B.1).

A consistent-counting reduction. Under a common denominator the executor grouping cancels and $\mathcal{M}$ reduces to the net log-potential drop rate; the implemented counting up-weights switch transitions without measurable effect on separation (Appendix A.2; Appendix C.2).

2.3 An In-Band Runtime Observer

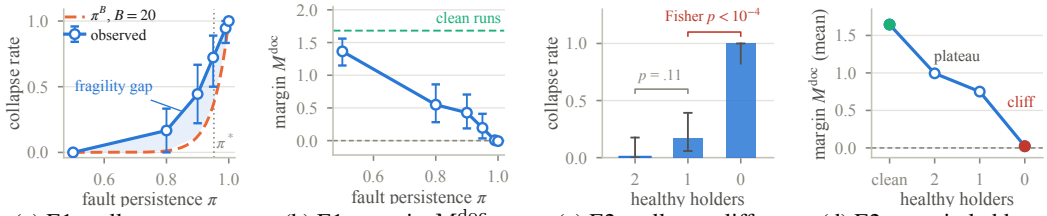
Gold checkpoints provide an offline reference for measuring task progress. The runtime observer uses the symbolic structure of the evaluated workflows: it extracts requested keys from the user query and marks a key as observed when executor traffic contains a corresponding key–value assertion. The fraction of requested keys supported in this way defines $\widehat{z}_k$; substitution into Equation (1) gives $\widehat{V}_k$ and the corresponding runtime margin. These estimates use only the in-band information $\mathcal{F}_k$, while benchmark checkpoints remain reserved for offline evaluation.

3 Controlled Destabilization and the Anatomy of Collapse

3.1 Controlled destabilization

To understand how orchestration degrades under tool failures, we construct a controlled fault-injection environment that separates the effects of model behavior, task difficulty, and tool reliability. Each task-relevant information item is assigned to one or more holders. When queried for that item, a holder returns its predefined value according to a deterministic request-response protocol.

We then vary the fault process along five controlled dimensions: target, persistence π, onset, number of failed replicas, and fault type (rate limit, silence, or silent corruption). For each task instance, the



(a) E1: collapse vs. π (b) E1: margin M^{doc} (c) E2: collapse cliff (d) E2: margin ladder

Figure 2: Two routes to collapse. (a, b) E1: as fault persistence π grows, collapse rises gradually above the retry-until-budget reference π^B and M^{doc} falls toward zero (95% intervals). (c, d) E2: collapse jumps only when the last healthy holder is removed, while M^{doc} already declines step by step.

clean and faulted conditions use the same dependency structure and underlying values, differing only in the injected fault configuration.

Before evaluation, a deterministic zero-LLM oracle validates each generated task. It verifies clean solvability, confirms that the target information becomes inaccessible when all relevant holders fail, and checks that the generated task contains no value leakage or identifier collisions. The oracle is used only for task validation; during execution, the observer and Governor have access only to the user query and executor traffic. We store all trajectories using the IWG bench/trajectory schema (Zhu et al., 2026), allowing the same parser and analysis pipeline to process controlled, external, and held-out evaluations. Experimental configurations and dataset details are reported in Appendix B.

3.2 The stochastic axis is a slope

The controlled destabilization in Section 3.1 considers two complementary failure regimes. The first is *stochastic failure*, where the required information remains available but repeated attempts to access it may fail with some probability. This models transient failures such as temporary rate limits or intermittent tool unavailability, where another retry can still succeed. The second is *structural failure*, where the final available information path disappears and the task can no longer be completed.

We study the stochastic regime by fixing the task structure and varying only the fault persistence π. The required information is provided by a single holder, whose requests may fail transiently; larger π makes consecutive failures more persistent and therefore recovery through retry less likely. We sweep $\pi \in \{0.5, 0.8, 0.9, 0.95, 0.99, 1.0\}$, with 18 faulted runs at each setting.

As fault persistence increases, the orchestrator collapses more frequently. Across $\pi \in \{0.5, 0.8, 0.9, 0.95, 0.99, 1.0\}$, the observed collapse rates under 18 runs are 0, 0.17, 0.44, 0.72, 0.94, and 1.0, respectively (Figure 2(a); M^{doc} in Figure 2(b)). To determine whether this increase is gradual or threshold-like, we fit three Bernoulli models to the binary collapse outcomes: a linear ramp, a logistic curve, and a change-point model. We use the Akaike information criterion (AIC) (Akaike, 1974) gives the linear-ramp model an AIC 22.7 points lower than the change-point model, supporting a gradual degradation pattern rather than a sharp threshold.

Having established the gradual degradation pattern, we next ask how much of the observed collapse can be explained by the stochastic fault process itself, and how much is associated with the orchestrator's persistence behavior. For each fault persistence π, we construct a simple retry-until-budget reference that never gives up early and uses all B available retry opportunities. Under the injected fault model, each attempt fails with probability π, so this reference collapses only when all B attempts fail, with probability π^B. We then compare this theoretical collapse rate with the collapse rate observed in the actual LLM orchestration runs.

When the observed collapse rate exceeds π^B, the orchestrator fails more often than the retry-until-budget reference under the same fault condition. We define this difference as the *behavioral fragility gap*, fragility gap $= p_{\mathrm{obs}}(\text{collapse}) - \pi^B$, where $p_{\mathrm{obs}}(\text{collapse})$ denotes the observed collapse rate of the LLM orchestrator under the corresponding fault condition. A larger gap means that the orchestrator fails more often than the retry-until-budget reference under the same fault condition and indicates that more potential recovery opportunities are left unused relative to the reference. The gap is substantial at intermediate persistence levels. For example, at $\pi = 0.8$, the retry-until-budget

reference predicts a collapse rate of only 1.2%, whereas the observed orchestrator collapses in 16.7% of runs.

An independently sampled Qwen3.8 Q8 replication shows the same qualitative pattern (Appendix B).

3.3 The Structural Axis Is a Cliff

We next study the second failure regime: *structural failure*, where available information paths are progressively removed. We ask how task failure changes as healthy holders are removed: whether the failure rate increases gradually, or jumps sharply when the last healthy holder is lost. To isolate this structural effect, we fix the fault persistence at $\pi = 1$ and vary the number of healthy holders that can still provide the required target information.

With two, one, and zero healthy holders, the observed collapse rates are 0%, 16.7%, and 100%, respectively (Figure 2(c)). The change from two to one healthy holder is limited (Fisher exact $p = 0.11$), while removing the final healthy holder produces a sharp jump to complete collapse (Fisher exact $p < 10^{-4}$). This transition is the solvable–unsolvable boundary fixed by the generator (Section 3.1). As long as at least one healthy holder remains, the required information can still be obtained. Once the final healthy holder is lost, the information becomes inaccessible and the task can no longer be completed. The collapse at zero healthy holders is therefore a property of the task construction; the behavioral result is the $3/18$ collapses with one healthy holder, where a viable path remains but the run still fails.

The completed-trajectory diagnostic M^{doc} (Section 2) provides an earlier view of this structural degradation. Its mean decreases from 1.645 in the clean condition to 0.995, 0.751, and 0.022 as the number of healthy holders decreases from two to zero (Figure 2(d)). Thus, while the final collapse rate changes sharply only when the last healthy holder disappears, the margin already reflects the gradual loss of progress before that boundary.

3.4 Local Failure Precedes Trajectory-Level Stall

We next ask when a local tool failure becomes a trajectory-level stall. A failed executor call alone does not imply that the overall task has stopped making progress, because other branches may still produce useful information. To examine this distinction, we inject the same fault at three positions in the dependency structure: leaf, terminal, and merge.

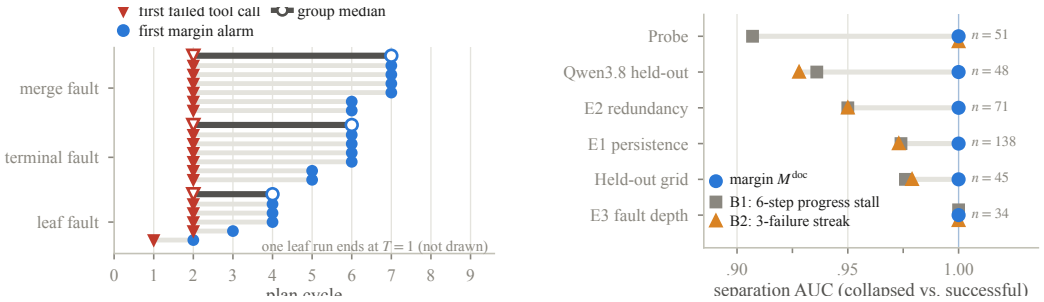
The first tool failure occurs at approximately step 2 in all three conditions. However, the trajectory-level alarm ($\tau_{\mathrm{on}} = 0$) appears later: at median steps 4, 6, and 7 for leaf, terminal, and merge faults, respectively. This delay shows that a local tool failure can precede the loss of trajectory-level progress by several steps (Figure 3(a)).

4 Validating the Runtime Margin

We first test whether the margin of Section 2 can distinguish trajectories that successfully complete the task from those that collapse. To quantify this ability, we use the area under the receiver operating characteristic curve (AUC; Fawcett, 2006), which measures how well a score separates the two groups across different decision thresholds. On a controlled probe of 63 trajectories, its completed-trajectory form M^{doc} achieves an AUC of 1.000, indicating complete separation between successful and collapsed trajectories on this dataset. This is retrospective discrimination on completed trajectories; it does not by itself predict recovery at a decision point.

An ablation shows that the separation is driven primarily by the progress term $w_u(1 - z_k)$. Removing this term reduces the AUC from 1.000 to 0.101, whereas removing any one of the remaining components-the failure term $w_f F_k$, the context-load term $w_c C_k$, or the repetition term $w_\ell R_k$-causes only a small decrease, with the AUC remaining close to 1.000. These results indicate that observed task progress z_k is the dominant signal for distinguishing successful from collapsed trajectories on this probe.

Does the margin outperform simpler rules? Having established that the margin can distinguish successful from collapsed trajectories, we next ask whether simpler runtime rules are sufficient for



(a) E3: local failure vs. first alarm (b) Margin vs. trivial monitors (AUC)

Figure 3: (a) A local tool failure precedes the trajectory-level stall: each dumbbell is one collapsed E3 run, from its first failed tool call (red) to its first margin alarm (blue, $\tau_{\text{on}} = 0$); the gap widens with fault depth. (b) Separation AUC of M^{doc} and two simple monitors on common eligible subsets (Table 6, Q1): M^{doc} reaches 1.000 on all six sets, B1 falls below on five and B2 on four; paired differences exclude zero only on E1 and the Qwen3.8 held-out grid.

detecting stalls. We compare the margin with two lightweight baselines (Figure 3(b)): B1 raises an alarm after six steps without new progress, while B2 raises an alarm after three consecutive failures on the same executor. On the collapsed E3 trajectories at $\tau_{\text{on}} = 0$, the margin detects $17/18$ runs, compared with $12/18$ for B1, and typically alarms one step earlier; B2 also detects $17/18$ and often reacts earlier than the margin. On recoverable successful trajectories, the margin alarms on $7/60$ runs at $\tau = -0.02$, against $6/60$ for B1 and $10/60$ for B2 (the margin–B2 difference interval includes zero); at $\tau_{\text{on}} = 0$ it alarms on $10/60$, matching B2 and exceeding B1 ($+0.067$ $[0.013, 0.156]$). No single operating point therefore makes the margin dominate both rules: it covers more stalls and warns earlier than B1 at the cost of more alarms on recoverable runs, while B2 reacts faster at comparable alarm rates. What distinguishes the margin is its graded response: along the E2 redundancy ladder M^{doc} changes at every level, whereas the B1 stall length and the B2 streak change only when the last healthy holder is removed (Appendix C.2).

Does the margin generalize to held-out tasks or models? Finally, we ask whether the runtime margin remains effective when the task configuration and deployment change, without retuning its parameters. We therefore keep the window fixed at $w = 6$ and the alarm threshold at $\tau = -0.02$. On the original probe, the resulting alarm detects all collapsed trajectories with a 6.5% false-alarm rate. We then evaluate the same frozen monitor on a held-out 48-run grid that changes task depth, width, fault type, distractor type, and fault persistence. The margin retains strong separation (M^{doc} AUC 0.991) with a 5.7% false-alarm rate, and detects all ten dynamical failures that provide enough trajectory history for the monitor to become active. A separate Qwen3.8 replication shows the same pattern: M^{doc} and the runtime margin both achieve AUC 1.000, and the alarm detects all eight collapsed trajectories without retuning.

5 DWELL: Adaptive Dwell-Time Governance

Sections 2-4 establish how trajectory-level progress can be measured and when such measurements are informative. Once a stall warning is available, the next question is *how the orchestrator should respond: should it continue retrying, or should it begin to wrap up?* This decision and intervention layer is referred to as *runtime governance*. Its role is to regulate how long the orchestrator continues expending resources after repeated failures. A stalled trajectory may still recover, whereas continued retries on an unsolvable task only consume the remaining budget.

Section 5.1 derives this rule, and the following experiments evaluate the resulting patience-stop-loss trade-off.

5.1 From a fixed threshold to a dwell rule

A simple way to stop the orchestrator from retrying indefinitely is to end the run after a fixed number of consecutive failures. A simple three-failure rule illustrates the cost: it shortens unsolvable runs

but forfeits recoverable successes (Table 8, Appendix C.4). Thus, a short failure streak alone is not sufficient evidence that further retries are futile.

Consecutive failures do not necessarily mean that recovery is unlikely. For example, if each call fails with probability $\pi = 0.8$, then three failures in a row occur with probability $0.8^3 = 0.512$, even though a later retry may still succeed. Thus, the persist-or-wrap-up decision should depend not only on the number of failures already observed, but also on how many retry opportunities remain.

To quantify the remaining recovery opportunity, we use a simple probabilistic model of repeated target-call failures. Let k denote the number of consecutive failures observed so far and let m denote the number of calls remaining. Throughout Section 5 the call budget is measured in executor calls, distinct from the plan-cycle length T (Appendix B.1).

Conditional on an unknown per-call failure probability π, successive calls are assumed independent, and we place a uniform prior $\pi \sim \mathrm{Beta}(1, 1)$ over the first probability. After observing k consecutive failures, the posterior becomes $\pi \mid k \sim \mathrm{Beta}(k + 1, 1)$. The posterior probability of obtaining at least one successful call within the remaining m attempts is therefore

$$\Pr(\text{success within } m \mid k) = 1 - \mathbb{E}[\pi^m \mid k] = \frac{m}{k + m + 1}, \tag{4}$$

(Proposition 1, Appendix A). The recovery probability decreases as the observed failure streak k grows and increases when more retry opportunities m remain. It therefore provides a model-based index of how much recovery opportunity remains at the current decision point.

The posterior recovery probability can be converted into a retry boundary. Retrying is considered worthwhile while $\Pr(\text{success within } m \mid k) \geq 1/(1 + \kappa)$, where κ is a constant that controls the allowed patience. Substituting Equation (4) gives $\frac{m}{k+m+1} \geq \frac{1}{1+\kappa}$, which is equivalent to $k+1 \leq \kappa m$. Using $m = B_{\text{left}}$, this gives the ideal retry boundary $k + 1 \leq \kappa B_{\text{left}}$. The implemented DWELL rule adds one conservative failure count and therefore continues retrying while

$$k + 2 \leq \kappa B_{\text{left}}. \tag{5}$$

Once this condition is violated, the policy enters *wrap-up*. We freeze $\kappa = 0.5$ before evaluation. The exact posterior derivation and integer stopping boundary are given in Appendix A.1. Because the posterior depends on the history only through (k, m), the rule is a deterministic threshold schedule indexed by the remaining budget (Equation (6)); it does not estimate the fault persistence of an individual task. Operationally, DWELL prevents premature stopping while the retry condition (5) holds and asks the planner to wrap up once the stopping boundary is crossed. Figure 6 illustrates the resulting retry-to-wrap-up transition on an unsolvable trajectory, while Appendix C.3 gives the corresponding executor-call and plan-cycle accounting.

5.2 Four-arm attribution under sustained baseline persistence

A three-arm evaluation on Qwen3.6-27B–OFF with no governance, v1 with wrap-up after three consecutive failures, and DWELL–shows that the adaptive rule (5) cuts the median unsolvable trajectory length from 13 to 7 and the total faulted budget by 31% while reroutable and clean tasks receive no intervention and keep their success; a sweep over $\kappa \in \{0.3, 0.5, 0.7\}$ places $\kappa = 0.5$ at the preselected knee (Table 8 in Appendix C.4; Appendix C.5). Earlier stopping thus reduces wasted effort, but this does not reveal whether the adaptive rule adds anything beyond early stopping itself. To answer this question, we conduct a pre-registered four-arm comparison on a self-hosted Qwen3.8-27B FP16 deployment (Appendix B.1). The four arms isolate different stopping mechanisms: OFF applies no governance; *Fixed-7* wraps up after seven consecutive failures; *Budget* wraps up after the ninth executor call; and *Full* uses the adaptive DWELL rule (5).

The evaluation uses 80 frozen tasks: 40 fragile tasks with $\pi \in \{0.8, 0.9\}$, 20 unsolvable tasks, and 20 reroutable tasks. Each task is run once under both clean and faulted conditions in all four arms, giving $80 \times 2 \times 4 = 640$ trajectories; all 640 are valid. The hypotheses, task pairings, and statistical tests were fixed before the first trajectory was run (Table 1).

P1: Does DWELL stop unsolvable runs earlier? On the 20 unsolvable tasks, DWELL stops much earlier than OFF, saving a paired median of 18.5 executor calls and 18 plan cycles (Holm-adjusted $p = 5.72 \times 10^{-6}$ for both measures). We then compare the three governed rules (Fixed-7, Budget, and

Table 1: Pre-registered four-arm attribution on Qwen3.8-27B FP16 ($n = 640$): means over the 20 unsolvable faulted tasks, success/intervention counts, and token totals over each arm's 80 faulted trajectories (key [Q38F-ARM], Appendix B).

Arm	Unsolv. T	Unsolv. N_{call}	Fragile	Clean succ./int.	Rerout. succ./int.	Faulted tokens
OFF	25.0	30.65	40/40	80/80 / –	20/20 / –	4,356,862
Fixed-7	8.0	13.55	25/40	80/80 / 0	19/20 / 0	2,456,607
Budget	5.35	10.95	17/40	80/80 / 0	20/20 / 1	1,995,242
DWELL	6.35	11.90	21/40	80/80 / 0	20/20 / 0	2,159,735

DWELL) on the 15 pre-registered tasks where their stopping boundaries differ. Fixed-7 is the most patient: DWELL stops a median of 3 executor calls earlier. Budget is the most aggressive, stopping a median of 1 call earlier than DWELL (Holm-adjusted $p = 1.22 \times 10^{-4}$ for both comparisons).

P2: What recoverable success is lost by stopping early? We next ask whether earlier stopping sacrifices tasks that could still recover. On the 40 fragile faulted tasks, Fixed-7 solves $25/40$, DWELL solves $21/40$, and Budget solves $17/40$. OFF, which imposes no stopping ceiling, solves all $40/40$ tasks. Thus, more aggressive stopping preserves fewer recoverable successes. In particular, compared with OFF, DWELL loses 19 successes and gains none ($21/40$ versus $40/40$; two-sided exact McNemar $p = 3.81 \times 10^{-6}$). The pre-registered tests also do not establish that DWELL outperforms either Fixed-7 or Budget. These results show the cost of stop-loss: stopping earlier on unsolvable tasks can also terminate recoverable tasks before they succeed (Figure 8; Appendix C.6). Governance leaves clean and reroutable runs intact (P3; Appendix B.1).

What drives the interventions? A post-hoc replay of all 1,419 governed trajectories shows that removing the runtime margin $\hat{\mathcal{M}}_k$ and keeping only the three-failure trigger changes the intervention time on no faulted trajectory: timing is driven by the failure counter and the budget arithmetic in (5), while $\hat{\mathcal{M}}_k$ serves as the measurement of Sections 3–4 (Appendix C.2).

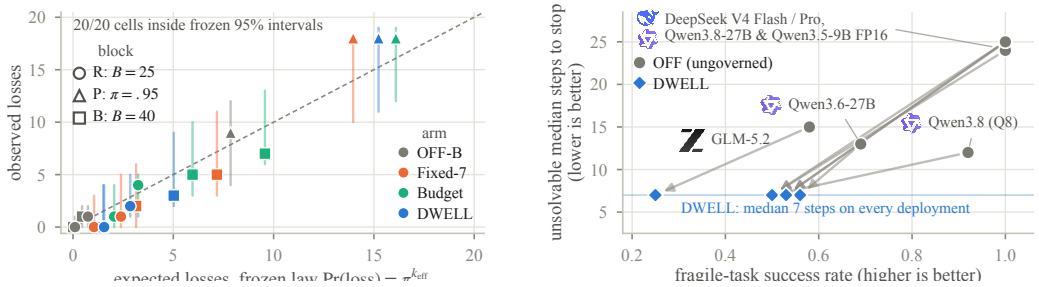
Does the patience-stop-loss trade-off transfer across models? Yes: on Qwen3.6-27B and on the complete Qwen3.5-9B matrix, DWELL again stops a paired median of 18 plan cycles earlier than OFF at a cost in fragile-task success (Appendix C.11; Appendix C.13).

Can the stopping law predict new conditions? A pre-registered epoch on 80 fresh tasks varies fault persistence and the executor-call budget shared by all arms. As an accounting check, the first wrap-up switch falls within one call of the rule's zero-LLM arithmetic in all 45 governed unsolvable cases (44 exact). All 20 pre-registered loss cells land inside their 95% intervals, although these are wide for small cells and fault seeds are shared across arms (Figure 4(a), Appendix C.14). The sharper test is task-level: every governed recoverable run is lost exactly when its seeded failure streak reaches the arm's threshold ($180/180$; OFF-B $59/60$, one early concession), and no run calls the target after wrap-up. Raising the budget from $B = 25$ to $B = 40$ moves DWELL's boundary by exactly five calls, and DWELL retains $27/30$ recoverable successes against $23/30$ for Fixed-7; on the same seeded streams, a budget-scaled fixed threshold (11 or 12 failures) retains the same $27/30$ (post-hoc), so this reversal reflects budget indexing. What transfers is behavioral: compliance makes each rule's recoverable-success cost a function of its threshold.

Does DWELL deliver more success per token? Exploratory token accounting gives DWELL/OFF success-per-token ratios above 1 by point estimate in all three epochs, less clearly for Qwen3.6 and not consistently against the fixed rules (Appendix C.12, Figure 10).

5.3 Baseline Persistence Shapes Governance Value

The evaluations above use deployments whose ungoverned orchestrator is already persistent. *oes the same frozen governance have the same value when native persistence differs?* Table 9 (Appendix C.4) collects seven evaluations under frozen task, parser, and Governor parameters; the faulted-only Q8 matrix and the FP16 epochs remain separate statistical datasets (Appendix B).



(a) Transfer epoch: predicted vs. observed losses (b) One frozen rule, seven deployments

Figure 4: (a) Pre-registered transfer epoch: for each of the twenty (block $\times$ π $\times$ arm) cells, recoverable-task losses expected under the frozen law $\Pr(\text{loss}) = \pi^{k_{\text{eff}}}$ against the observed losses, with frozen 95% prediction intervals; all $20/20$ fall inside (criterion: at least 18). Intervals are wide for the $n = 5$ replication cells, and fault seeds are shared across arms. (b) The same frozen DWELL rule on seven deployments: each arrow runs from the ungoverned point (OFF) to the DWELL point (Table 9). Stop-loss is portable, a seven-step median everywhere, while the recoverable success retained depends on the deployment.

The ungoverned baselines differ widely. OFF fragile-task success ranges from 0.58 (GLM-5.2) to 1.00 (both DeepSeek V4 deployments and the FP16 epochs), and median unsolvable length from 12 to 25 steps: some orchestrators persist through recoverable faults, others concede early. The fixed three-failure rule ignores this variation. It caps fragile success at 0.33 across all five three-arm evaluations, set by the shared frozen fault streams rather than the orchestrator, so its cost depends on deployment: the same ceiling forfeits 0.25 of GLM-5.2's recoverable successes but 0.67 of DeepSeek V4's.

DWELL shows the same deployment dependence (Figure 4(b)). Its stop-loss is portable: the unsolvable-task median is seven steps in all seven evaluations (significant for Qwen3.6-27B and both DeepSeek deployments; the GLM-5.2 and Q8 comparisons are underpowered). Fragile-task success ranges from 0.25 to 0.56, so the same rule lands at a different point on the patience–stop-loss frontier under each orchestrator. Where baseline persistence is high, a stopping ceiling trades recoverable successes for savings; where it is low, a fragility gap leaves room to improve both. The value of a stopping rule thus belongs to the deployment–rule pair, and the fragility gap of Section 3 locates it on this map. Clean runs retain success wherever controls were run (the single GLM-5.2 difference reflects trajectory stochasticity, with no recorded Governor action), and v1/v2 interventions on reroutable tasks are zero across the three-arm evaluations. Per-model tests, intervention attribution, provider-epoch caveats, and the excluded K3 batch are reported in Appendix C.4.

6 Discussion and Conclusion

What the experiments add beyond the rule. The rule decides when to ask the planner to wrap up; the experiments measure how the planner responds. In audited runs, planners retry until success or the threshold, then make no further target calls. Recoverable tasks are therefore lost when recovery would occur after the threshold, allowing ungoverned logs to estimate this cost. The cost varies by deployment: OFF success on recoverable tasks ranges from 0.58 to 1.00, while DWELL stops unsolvable runs at a median of seven steps in all seven evaluations. Under independent faults, DWELL uses only the failure count k and remaining calls m, making it equivalent to a budget-scaled fixed threshold. More structured faults are needed to demonstrate additional adaptive value.

Conclusion. We introduced a stability margin with a runtime observer and DWELL, a controller allocating retries from failure history and remaining budget. Controlled experiments reveal two routes to failure and differences in persistence across deployments. four-arm study and two replications, DWELL saves a paired median of 18 planning cycles on unsolvable tasks versus OFF. Fresh-task tests link recoverable-success losses to stopping thresholds. Governance should therefore be evaluated by resources saved and tasks completed.

AI USE STATEMENT

Generative AI tools were used to assist with drafting and language editing. All technical claims, experiments, citations, and final wording must be verified and approved by the authors before submission.

REPRODUCIBILITY STATEMENT

Appendix A states the assumptions and provides complete proofs; Appendix B records dataset counts, frozen configurations, exclusions, deployment epochs, hashes, and source locations; and Appendix C reports supplementary analyses. The anonymized supplement contains the manuscript sources, figure assets and generator, analysis code, tests, frozen reports, and raw valid trajectories under an explicit file allowlist, together with commands and checksums for verification.

REFERENCES

Hirotugu Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974. doi: 10.1109/TAC.1974.1100705.

Rahul Suresh Babu and Adarsh Agrawal. Self-healing agentic orchestrators for reliable tool-augmented large language model systems. *arXiv preprint arXiv:2606.01416*, 2026.

Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? *arXiv preprint arXiv:2503.13657*, 2025.

Chaoran Chen, Vy Nguyen, Ziji Zhang, Abhinav Gullapalli, Ziyi Wang, Yuxuan Lu, Dakuo Wang, Jing Huang, Zhou Yu, and Jin Lai. Retry, switch, or abstain? learning strategy-aware tool-use policies via controlled error injection. *arXiv preprint arXiv:2608.11977*, 2026. doi: 10.48550/arXiv.2608.11977.

Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. doi: 10.1016/j.patrec.2005.10.010.

Romain Froger, Pierre Andrews, Matteo Bettini, Amar Budhiraja, Ricardo Silveira Cabral, Virginie Do, Emilien Garreau, Jean-Baptiste Gaya, Hugo Laurençon, Maxime Lecanu, Kunal Malkan, Dheeraj Mekala, Pierre Ménard, Gerard Moreno-Torres Bertran, Ulyana Piterbarg, Mikhail Plekhanov, Mathieu Rita, Andrey Rusakov, Vladislav Vorotilov, Mengjue Wang, Ian Yu, Amine Benhalloum, Grégoire Mialon, and Thomas Scialom. ARE: Scaling up agent environments and evaluations. *arXiv preprint arXiv:2509.17158*, 2025.

Aayush Gupta. ReliabilityBench: Evaluating LLM agent reliability under production-like stress conditions. *arXiv preprint arXiv:2601.06112*, 2026.

Nicholas Hay, Stuart Russell, David Tolpin, and Solomon Eyal Shimony. Selecting computations: Theory and applications. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, pp. 346–355. AUAI Press, 2012.

João P. Hespanha and A. Stephen Morse. Stability of switched systems with average dwell-time. In *Proceedings of the 38th IEEE Conference on Decision and Control*, pp. 2655–2660, 1999. doi: 10.1109/CDC.1999.831330.

Daniel Liberzon. *Switching in Systems and Control*. Birkhäuser, 2003.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Yuxiang Lin, Zihan Wang, Mengyang Liu, Yuxuan Shan, Longju Bai, Junyao Zhang, Xing Jin, Boshan Chen, Jinyan Su, Xingyao Wang, Jiaxin Pei, and Manling Li. BAGEN: Are LLM agents budget-aware? *arXiv preprint arXiv:2606.00198*, 2026. doi: 10.48550/arXiv.2606.00198.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024.

Michael Luby, Alistair Sinclair, and David Zuckerman. Optimal speedup of Las Vegas algorithms. *Information Processing Letters*, 47(4):173–180, 1993. doi: 10.1016/0020-0190(93)90029-9.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An analytical evaluation board of multi-turn LLM agents. *arXiv preprint arXiv:2401.13178*, 2024.

Kerri Prinos, Lilianne Brush, Cameron Denton, Zhanqi Wang, Joshua Knox, Snehal Antani, Anton Foltz, and Amy Villaseñor. Stable agentic control: Tool-mediated LLM architecture for autonomous cyber defense. *arXiv preprint arXiv:2605.03034*, 2026.

Kai Ruan, Zihe Huang, Ziqi Zhou, Qianshan Wei, Jinghao Lin, Xuan Wang, and Hao Sun. Doomed from the start: Early abort of LLM agent episodes via a recall-controlled probe cascade. *arXiv preprint arXiv:2607.06503*, 2026. doi: 10.48550/arXiv.2607.06503.

Stuart Russell and Eric Wefald. Principles of metareasoning. *Artificial Intelligence*, 49(1–3):361–395, 1991. doi: 10.1016/0004-3702(91)90015-C.

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.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Junze Zhu, Weihao Chen, Xuanwang Zhang, Zhen Wu, and Xinyu Dai. Recognize your orchestrator: An entropy dynamics perspective for LLM multi-agent systems. *arXiv preprint arXiv:2606.01351*, 2026.

A Formal Details

A.1 Posterior recoverability and exact Governor accounting

Proposition 1 (Posterior threshold with a one-count guard)**.** *Under A3 and a uniform prior, after k failures the probability of at least one success in the next m calls is $p_{k,m} = m/(k+m+1)$. The ideal rule $p_{k,m} \geq 1/(1+\kappa)$ is equivalent to $k+1 \leq \kappa m$. The implemented rule $k+2 \leq \kappa m$ is therefore an exact one-count conservative refinement. At its equality boundary,*

$$p_{k,m} - \frac{1}{1+\kappa} = \frac{1}{(1+\kappa)(k+m+1)} = O(m^{-1}).$$

Proof. The posterior is $\pi \mid k \sim \mathrm{Beta}(k+1, 1)$, and hence

$$1 - \mathbb{E}[\pi^m \mid k] = 1 - (k+1)\int_0^1 \pi^{k+m}\, d\pi = \frac{m}{k+m+1}.$$

Cross multiplication gives the ideal count boundary and the displayed gap. The extra count in the code is motivated by the plug-in posterior mean $(k+1)/(k+2)$; it is not an unquantified approximation. □

Proposition 2 (Frozen stop-loss accounting)**.** *Let c_0 be executor calls charged to the budget but not belonging to the permanently unavailable target's failure streak. After k target failures, the implementation sees $m = B - c_0 - k$ calls left and first enters wrap-up at*

$$k_{\mathrm{wrap}} = \left\lfloor \frac{\kappa(B - c_0) - 2}{1+\kappa} \right\rfloor + 1. \tag{6}$$

For $B = 25$, $c_0 = 2$, and $\kappa = .5$, this is $k_{\text{wrap}} = 7$ and the switch occurs after $N_{\text{call}} = c_0 + k_{\text{wrap}} = 9$ executor calls. With the frozen hard trigger three and escalation interval two, L3 is also first available after the seventh target failure. If the final order is obeyed, it adds one planner call and no executor call or tool-bearing cycle; if it is ignored, the forced-stop guard permits at most one further tool-bearing cycle before the next planner call stops the run.

Proof. Wrap-up is selected exactly when $k + 2 > \kappa(B - c_0 - k)$. Equation (6) is the smallest integer satisfying this strict inequality. Thus the often used ceiling expression is equivalent only when its argument is non-integral. With the frozen values, the sixth failure leaves $m = 17$ and satisfies $8 \le 8.5$, whereas the seventh leaves $m = 16$ and satisfies $9 > 8$.

The hard alarm begins at failure three. Thereafter its uncapped run length is $r = k - 2$, and the ladder level is $\min\{3, 1 + \lfloor (r-1)/2 \rfloor\}$, so L3 is first reached at $k = 7$. Retry mode caps the active level at L1; the cap is removed on the same tick that selects wrap-up. The L3 branch sets the final-order latch before calling the planner, and a subsequent planner call returns the forced-stop object. □

The six primary trajectories cited in the original mechanism audit run two independent branches concurrently. Their healthy branch has two serial calls and their unavailable branch seven, while the final order executes no tool. Consequently their stored planner-cycle length is

$$T = \max\{2, 7\} + 0 = 7,$$

not $c_0 + k_{\text{wrap}} = 9$. The latter is the call-budget count. All six have action counts $(2, 2, 1, 1, 1, 1, 1)$, retry advisories at failure counts $3, 4, 5, 6$, then an L3 wrap-up tick and one obeyed final order. File-level hashes and the read-only extraction commands are retained in the repository's WP-B2 accounting audit.

A.2 Hard-stall alarm

Scope of the runtime observer. This observer measures transcript-supported completion of explicit query requirements. Its application depends on an appropriate progress representation for the workflow. Semantic correctness requires value-level evidence, as examined by the corruption experiments in Section 4. The hard-stall guarantee below applies to the specified potential, update conditions, and window convention.

Lemma 1 (Exact-runtime hard-stall bound)**.** *Fix EMA retention $a = .6$, weights $(1, .4, .2, .6)$, window $w = 6$, and the implemented per-transition margin in Equation (3). Suppose a hard-stall segment is preceded by an effective update, contains six valid potential transitions, has constant progress, and has nondecreasing bounded context load. Then the final full window satisfies*

$$M_6 \le -.0213987587 < \tau = -.02.$$

Thus the alarm fires by the sixth hard-stall update, with a worst-case runtime margin of at least .0013987587 beyond the frozen threshold.

Proof. The unused failure/repetition headroom before the segment is at least $(w_f + w_\ell)(1 - a) = .4$. Against the maximum potential 2.2, the six successive log-growth terms have the conservative lower bounds

$$(g_1, \ldots, g_6) \ge (.0851578, .0478179, .0276303, .0162190, .00960662, .00571998).$$

The first transition is classified as within-mode. For any set J of $n > 0$ switches among the remaining five transitions, the growth side of the implemented margin is

$$-M_6 = \frac{5 - n}{5(6 - n)} \sum_{j \notin J} g_j + \frac{1}{5} \sum_{j \in J} g_j;$$

for $n = 0$ it is $\sum_j g_j / 6$. Direct evaluation of the 2^5 switch masks places the minimum at $n = 5$, where

$$-M_6 \ge \frac{1}{5} \sum_{j=2}^{6} g_j = \frac{1}{5} \log \frac{V_6}{V_1} = .0213987587.$$

All discarded context increments are nonnegative, so they cannot weaken the bound under the stated assumptions. □

This exact-runtime constant supersedes the stronger .017 slack obtained by an earlier five-ratio indexing argument. It is deliberately not transferred to mixed stalls: its remaining runtime slack is only about .00140. The first-transition and switch-count conventions used in this proof are the implemented counting behind every reported margin quantity; Section 2 states the consistent-counting reduction under which the executor grouping cancels.

A consistent-counting reduction. If the switch share used the same denominator as the two group means, the executor grouping would cancel and $\mathcal{M}$ would reduce to the net log-potential drop rate. The implementation instead classifies each window's first transition as within-mode and counts the switch share over the remaining transitions, which up-weights switch transitions (counting convention below); on the evaluated sets this adds no measurable separation (net drop rate AUC $1.000/0.987/0.989$ on the probe, held-out, and E1 sets, against $0.968/0.989/0.990$ for $\mathcal{M}$; Appendix C.2). We therefore read $\overline{\log \rho}$ and $\overline{\log \mu}$ as descriptive diagnostics; all reported quantities use the implemented counting.

Counting convention. If a window contributes N usable transitions and the switch share uses the same denominator as the two group means ($\nu = N_{\text{switch}}/N$), the weighted sum in Equation (3) telescopes into the window's net log-potential drop rate,

$$\mathcal{M} = -\frac{1}{N}\sum_j g_j = -\frac{1}{N}\log\frac{V_T \vee \epsilon}{V_{T-N} \vee \epsilon},$$

and the executor grouping cancels in the aggregate. The implementation departs from this consistent counting: the first transition of a window is classified as within-mode and the switch share is counted over the remaining five, so in a six-step window with $n > 0$ switches each switch transition carries weight $1/5$ against each within-mode transition's $(5-n)/(5(6-n))$, and with $n = 0$ all six weights equal $1/6$—the decomposition used in the proof of Lemma 1. The weighting ablation in Appendix C.2 compares the two countings on the frozen trajectories.

A.3 Mixed-stall detectability domain

Consider the explicit subclass in which progress is fixed and exactly one of the failure and repetition indicators is active at each update. Let $q_j = .4F_j + .6R_j$ and let $p_j \in \{.4, .6\}$ select failure or repetition. Then

$$q_j = .6q_{j-1} + .4p_j, \qquad q_m = .6^m q_0 + D(x_{1:m}), \quad D = .4\sum_{j=1}^{m} .6^{m-j}p_j. \tag{7}$$

Proposition 3 (Explicit mixed-stall window domain)**.** *For a same-executor window, write the non-risk part of the initial potential as C_0, its window increment as $\delta_C \geq 0$, and $\gamma = -\tau$. The window alarms exactly when*

$$q_0 < Q_\gamma(x_{1:m}, C_0, \delta_C) := \frac{D(x_{1:m}) + \delta_C - (e^{m\gamma} - 1)C_0}{e^{m\gamma} - .6^m}. \tag{8}$$

Equality does not alarm because the implementation uses a strict threshold. With executor switches, let $L_j = \log(V_j/V_{j-1})$, let n of the final $m-1$ transitions switch, and let S_s, S_w sum the switch and within-mode terms. The exact growth score is

$$G_m = -M_m = \begin{cases} (S_w + S_s)/m, & n = 0, \\ \dfrac{m-1-n}{(m-1)(m-n)}S_w + \dfrac{1}{m-1}S_s, & 1 \leq n \leq m-1. \end{cases}$$

Hence detection depends jointly on initial EMA risk, the indicator word, window phase, and executor-switch phase, rather than only on the ordinary fraction of repeat steps.

Proof. Substitute Equation (7) into $V_m > e^{m\gamma}V_0$ and solve for q_0 to obtain Equation (8). Separating the implemented weighted mean into its switch and within-mode terms gives the second display. □

The limitation is structural, not an invitation to retune τ. Strict failure/repetition alternation has the period-two risk orbit $q_L = .475, q_H = .525$. A same-executor six-step window on this orbit has

$G_6 = 0$ and alarms at neither the online nor offline threshold. With switches, phase can make the online score positive or negative; for unresolved baseline $C \geq .1$, even its most favorable switch phase has $G_6 \leq .016676 < .02$ and therefore cannot cross the offline threshold. Transient alternating stalls, when detectable, first have enough samples at three updates and cross within three to six updates; if the first full window does not cross, no later finite-delay guarantee follows. These claims concern the stated single-indicator subclass, not arbitrary mixed failures or global black-box LLM stability.

A.4 A CLASSIFICATION OF IN-BAND DETECTABILITY

Let $\mathcal{S}_{w,m}$ contain partial, $\mathcal{F}_k$-measurable statistics that depend only on the latest w updates and output $\perp$ whenever fewer than m valid transitions are available. Let $N_{k,w}$ be the valid count in the current window and N_k^{cum} the monotone cumulative valid count. This notation is distinct from the context-load term C_k in Equation (1).

Proposition 4 (Three constructed failure families)**.** *For $\mathcal{S}_{6,3}$, the following scoped statements hold.*

1. *Under Lemma 1's assumptions, the implemented margin is a constructive member of $\mathcal{S}_{6,3}$ that detects the stated hard-stall family within one window.*
2. *If a run terminates with $N_T^{\text{cum}} < m$, no member of $\mathcal{S}_{w,m}$ can raise a numeric-statistic alarm on that run. This is a lower bound for this window-statistic class, not for every in-band observer.*
3. *Let W_1 have true terminal value v, and let W_2 have true value $v' \neq v$ but a poisoned terminal holder that emits v. If the complete in-band transcripts have the same distribution in both worlds, no $\mathcal{F}_T$-measurable observer distinguishes them beyond the prior; under equal priors, binary accuracy is at most $1/2$.*

Proof. The first claim is Lemma 1. For the second, monotonicity gives $N_{k,w} \leq N_k^{\text{cum}} \leq N_T^{\text{cum}} < m$ for every $k \leq T$; the minimum-sample gate therefore forces every statistic output to $\perp$. For the third, write the full transcript as X_T. Equal laws for X_T imply equal laws for every measurable function of X_T and independent observer randomness. Thus an equal-prior classifier's true-positive probability in one world equals its false-positive probability in the other, yielding accuracy $1/2$. □

The three instruments in this paper should therefore be read as *coverage-complementary* on these constructed families: trajectory dynamics cover the stated hard stalls, decision-point statistics can provide a signal before a window forms, and an out-of-band truth anchor or a protocol-enforced cross-source constraint is required for the equal-transcript poisoning construction. Except for that equal-transcript information bound, this is not a theorem that the channels are universally necessary, orthogonal, irreplaceable, or a minimal sufficient basis. The construction does not exhaust orchestration failures, and mid-chain poisoning that changes the downstream transcript is outside the third claim.

A.5 NOTATION AND TERMINOLOGY

B EXPERIMENTAL PROTOCOL

The rows in the table sum to $63 + 144 + 72 + 35 + 36 + 240 + 252 + 48 + 48 + 18 + 36 + 54 + 72 + 48 + 640 + 612 + 640 + 640 = 3{,}698$ valid trajectories. The three-arm Governor subtotal remains $240 + 252 + 72 = 564$. The primary four-arm matrix contributes 640 valid trajectories; the separate Qwen3.6 FP16 comparison contributes 612 valid trajectories from 640 attempted cells. Its 28 contract-invalid cells and nine interrupted-block records are audited separately (Appendix C.11). The Qwen3.5-9B FP16 epoch contributes 640 valid trajectories from 640 attempted cells (Appendix C.13). The pre-registered transfer epoch contributes 640 valid trajectories from 640 attempted cells under a unified harness call cap (Appendix C.14); its capped OFF-B arm is reported separately from the uncapped OFF of earlier epochs. The epochs and the 72-run Q8 Governor matrix remain separate statistical datasets. These totals exclude exploratory smoke runs, capability gates, contract tests, the

Table 2: Frozen notation and terminology. Code names are retained where they remove ambiguity between the reported diagnostic and the runtime score.

Term	Symbol / code	Meaning
margin	$\mathcal{M}$, `margin_rate`	per-transition contraction score
average-dwell diagnostic	M^{doc}, `margin_doc`	reported alternative weighting
dwell	run length / `mean_dwell`	consecutive time in one executor mode
regime	`retry`, `wrapup`	Governor control state
terrain	fragile, unsolvable, reroutable, clean	evaluation-world category
mode	σ_k / executor	active tool holder
window count	$N_{k,w}$	valid transitions in the latest window
cumulative count	N_k^{cum}	monotone valid-transition count
context load	C_k	capped context term in the potential

Table 3: Generator controls and the questions they isolate.

Control	Sweep	Scientific question
Persistence	$\pi \in [0.5, 1]$	gradual degradation or cliff?
Redundancy	healthy holders $2, 1, 0$	where is the structural boundary?
Fault depth	leaf / terminal / merge	error time or stall time?
Fault type	failure / silence / poison	what can an observer detect?

Qwen3.8 full Probe, and invalid attempts; the Q8 Qwen3.8 increment is $54 + 72 + 48 = 174$, and the four FP16 epochs contribute $640 + 612 + 640 + 640 = 2{,}532$ valid trajectories.

Source ledger. Table 5 resolves each bracketed dataset key into its raw-data directory and audit record.

All three Qwen3.8 source keys share model `qwen38-27b-a`; the deployment audit is `phase0/QWEN38_27B.md` (SHA-256 `984d1c8b...2216`). The report-level audits are `phase0/results_qwen38_27b_e1_reduced/e1_report.md`, `phase0/results_gov_qwen38_27b/qwen38_governor_audit.md`, and `phase0/results_holdout_qwen38_27b/qwen38_holdout_audit.md`.

The controlled generator stores task specification, seed, topology, holders, fault parameters, and expected answer. The orchestrator receives only the query and executor interfaces. Outcome success is determined from the final answer; checkpoint coverage drives gold progress but is not itself the outcome label. All main margin analyses use order-free cumulative checkpoint matching. The frozen online configuration is $w = 6$, $\tau = -.02$; the Governor configuration is $B = 25$, $\kappa = .5$, cooldown length five, and a three-failure hard trigger. Invalid zero-step, partial, parsing-error, and quota-terminated files are isolated before analysis.

The three 84-run cross-model Governor replications use the same 24 faulted task specifications per arm; OFF and v2 additionally receive the same six clean controls. The Qwen3.8 replication uses the same 24 frozen faulted specifications per arm but deliberately contains no Governor clean controls, for 72 total trajectories. Its specification/bench manifest digests are `f1babcd4...aafa`/`30b09cfd...8a25`; source: `phase0/results_gov_qwen38_27b/`. Model aliases, API providers, and deployment epochs are retained as separate metadata; invalid attempts are never used to replace or inflate valid trajectory counts.

The Qwen3.8 Holdout uses 24 frozen task instances under both clean and faulted conditions. Its clean alarm count is $0/24$. The distinct $5/40$ quantity is the collapse detector's risk-alarm count over all successful trajectories; all five alarms occur on successful faulted trajectories, not clean runs. Source: `phase0/results_holdout_qwen38_27b/`.

Qwen3.8 Q8 replication of E1. The same qualitative pattern appears in an independently sampled Qwen3.8 Q8 replication: collapse increases from $1/9$ to $7/9$ to $9/9$ as π increases from 0.8 to 0.95 to 1.0. The linear-ramp model remains preferred, supporting the transfer of the gradual-degradation pattern across deployments.

Table 4: Core experimental datasets. Counts refer to valid stored trajectories; exploratory smoke runs are excluded from claims.

Dataset	n	Purpose	Src.
Controlled probe	63	margin separation	[P]
E1 persistence	144	stochastic slope	[E1]
E2 redundancy	72	structural cliff	[E23]
E3 fault depth	35	detection timing	[E23]
E4 corruption	36	observability boundary	[E4]
Primary Governor v1/v2/OFF	240	intervention effects	[G0]
Cross-model Governor replications	252	Flash, Pro, GLM	[GX]
Frozen holdout	48	topology/fault transfer	[H0]
IWG snapshot	48	external audit	[IWG]
K3 stage-2 probe	18	ranking transfer	[K3S]
K3 dynamic probe	36	runtime calibration	[K3D]
Qwen3.8 reduced E1	54	frozen E1 transfer	[Q38-E1]
Qwen3.8 Governor	72	faulted-only transfer	[Q38-G]
Qwen3.8 Holdout	48	frozen detector transfer	[Q38-H]
Qwen3.8 FP16 four-arm	640	mechanism attribution	[Q38F-ARM]
Qwen3.6 FP16 same-design	612	separate comparison epoch	[Q36F-ARM]
Qwen3.5-9B FP16 same-design	640	no-gap boundary epoch	[Q359B-ARM]
Qwen3.8 FP16 transfer (unified cap)	640	pre-registered prediction transfer	[TRANSFER-ARM]

Held-out depth-3 topology. A held-out depth-3 topology shows the same qualitative pattern: faulted runs continue to reroute successfully as long as a healthy information path remains available.

B.1 FOUR-ARM PROTOCOL (QWEN3.8-27B FP16)

Arms. All four arms share the frozen observer and Governor parameters ($w = 6$, $\tau = -.02$, online $\tau_{\text{on}} = 0$ with hysteresis $\tau_{\text{off}} = .05$, $\kappa = .5$, $B = 25$, hard trigger three, escalation interval two, cooldown five) and differ only in the regime-switch trigger. OFF instantiates no Governor. Fixed-7 enters wrap-up when a single target agent reaches its seventh consecutive failure and performs no budget arithmetic. Budget enters wrap-up when cumulative executor-wrapper calls reach nine and ignores streaks. Full is the frozen v2 rule: wrap-up iff $k + 2 > .5\, B_{\text{left}}$ with $B_{\text{left}} = 25 - N_{\text{call}}$ (frozen `governor.py`, lines 206–207). Retry-regime advisories, the wrap-up ladder, final order, cooldown, veto, parser, and prompts are shared verbatim by the three governed arms.

Tasks. The frozen manifest `phase0/ARM_ATTRIBUTION_TASKS_20260901.json` fixes 80 tasks with seeds $20260901000 + i$: 20 fragile at $\pi = .8$ and 20 at $\pi = .9$ (one holder), 20 unsolvable ($\pi = 1$, no healthy replica), and 20 reroutable ($\pi = 1$, three replicas, exactly one healthy). Within each family of 20 the (onset, c_0) strata are $(0, 2) \times 5$, $(0, 5) \times 3$, $(0, 10) \times 2$, $(5, 5) \times 3$, $(5, 10) \times 3$, and $(10, 10) \times 4$, so $c_0 \in \{2, 5, 10\}$ occurs $20/24/36$ times and onset $\in \{0, 5, 10\}$ occurs $40/24/16$ times over the 80 tasks; topology is depth one with width c_0 and the fault on the merge key. Each task runs once under clean and once under faulted conditions in every arm ($80 \times 2 \times 4 \times 1 = 640$). A zero-LLM oracle established before the run that Full and Fixed-7 diverge on $60/80$ tasks and Full and Budget on $60/80$ (admission threshold $25/80$); the 15 unsolvable tasks with $c_0 \in \{5, 10\}$ form the pre-registered oracle-divergent subset for the Fixed-7 and Budget contrasts.

Pairing, tests, and no-peeking. The pairing key is (task index, condition, trial). P1 consists of six one-sided paired Wilcoxon signed-rank tests with exact sign-flip enumeration and Holm correction over the six: Full<OFF on N_{call} and T over all 20 unsolvable tasks, and Full<Fixed-7 and Budget<Full on both clocks over the oracle-divergent 15 (the 20-task versions of the latter are sensitivity analyses only). P2 consists of exact one-sided McNemar tests (Full>Fixed-7, Full>Budget) with Holm correction over the two; Full vs. OFF is a two-sided harm screen outside the confirmatory family. P3 reports per-arm intervention proportions with rule-of-three upper bounds or Clopper–Pearson intervals. Confidence intervals are 10,000-sample paired bootstrap percentile intervals (seed 20260901). Non-significance is never read as equivalence or non-inferiority. Per the pre-registration, no arm-level success, length, token, or intervention aggregate was computed until all 640 trajectories were valid ($640/640$; zero invalid); the driver's progress record and structural audit are retained beside the results.

Table 5: Source ledger: raw-data directory, deployment epoch, and audit record for each bracketed dataset key used in the dataset census above.

Key	Raw-data directory	Deployment epoch and audit record
[P]	`phase0/results_probe/`	—
[E1]	`phase0/results_e1/`	—
[E23]	`phase0/results_e23/`	—
[E4]	`phase0/results_e4/`	—
[G0]	`phase0/results_gov/`	—
[GX]	`phase0/results_gov_deepseekv4flash_official/`; `phase0/results_gov_deepseekv4pro_aliyun/`; `phase0/results_gov_glm52_tokenplan/`	—
[H0]	`phase0/results_holdout/`	—
[IWG]	`phase0/results_iwg_revisit/`	—
[K3S]	`phase0/results_probe_k3_stage2/`	—
[K3D]	`phase0/results_probe_k3_e2_frozen/`	—
[Q38-E1]	`phase0/results_qwen38_27b_e1_reduced/`	shared epoch; see below
[Q38-G]	`phase0/results_gov_qwen38_27b_/`	shared epoch; see below
[Q38F-ARM]	`phase0/results_arm640_qwen38fp16/`	epoch `qwen38-27b-fp16-20260902-b`; pre-registration `phase0/ARM640_PREREGISTRATION_20260901.md` (SHA-256 `0675aff2...b973`); protocol in Appendix B.1
[Q36F-ARM]	`phase0/results_arm640_qwen36fp16b_20260913/`	epoch `qwen36-27b-fp16-20260907-b`; original addendum `phase0/ARM640_QWEN36_COMPARISON_ADDENDUM_20260907.md`; completed-data analysis in Appendix C.11
[Q359B-ARM]	`phase0/results_arm640_qwen359b_20260918/`	epoch `qwen35-9b-fp16-20260918-a`; original addendum `phase0/ARM640_QWEN35_9B_ADDENDUM_20260918.md`; completed-data analysis in Appendix C.13
[TRANSFER-ARM]	`phase0/results_transfer_20260920/` (transfer worktree, branch `transfer-20260920`)	epoch `qwen38-27b-fp16-20260920-t`; pre-registration `phase0/ARM640_TRANSFER_PREREGISTRATION_20260920.md` (SHA-256 `26dadceb...`); analysis in Appendix C.14

Fragility-gap baseline. Section 3 reports the original E1 baseline as π^B with a fixed $B = 20$; those numbers are unchanged. The four-arm pre-registration replaces the fixed constant by the per-topology oracle budget $B_{\text{target}} = 25 - c_0 \in \{15, 20, 23\}$, under which a budget-rational agent collapses only if B_{target} consecutive target retries fail. The stratified mean baseline over the 40 fragile tasks is .086 (.0208 at $\pi = .8$ and .1513 at $\pi = .9$), against $0/40$ observed OFF collapses (Clopper–Pearson $[0, .088]$; $0/20$ in each π stratum). The two baselines coincide only for $c_0 = 5$ and are reported separately, never mixed.

Budget clocks. Here and throughout Section 5, the call budget is measured in executor calls. We distinguish this from the trajectory length T, which counts planner cycles; a single planner cycle may contain multiple executor calls. Three quantities coexist by design and are never interchanged. (i) The harness imposes the same external hard cap on every arm: $T \leq 25$ plan cycles. (ii) Each

policy chooses its own stopping time under that cap: the Governor's call budget $B = 25$ executor calls constrains only the three governed arms, while OFF, as the ungoverned deployment, has no call cap and can exceed B (maxima among fragile-task successes: 30 calls on 3.8 and 9B, 27 on 3.6; on unsolvable tasks OFF averages 30.65 calls on 3.8 before the cycle cap ends the trajectory, Table 1). (iii) The fragility-gap baseline grants its retry-until-budget reference $B_{\text{target}} = 25 - c_0$ target opportunities—the retries available inside the Governor's budget—rather than OFF's realized call count. Because the reference collapses only if every opportunity fails, a reference granted OFF's larger realized spending would collapse less often than $\pi^{B_{\text{target}}}$; the reported baseline is thus an upper bound on that reference's collapse rate, and OFF's $0/40$ fragile collapses read as no detectable gap under either opportunity count. The pre-registered endpoints keep the OFF asymmetry deliberately: OFF is evaluated as the ungoverned deployment found in practice. The asymmetry inflates the raw stop-loss comparison in Full's favor (OFF's uncapped spending only enlarges Full's stop-loss margin) and favors OFF in the harm screen whenever OFF succeeds only beyond the governed budget. A budget-matched sensitivity for the affected contrasts is in Appendix C.7; in the fresh-task epoch (Appendix C.14) the harness cap applies to every arm (OFF-B), and the asymmetry is absent by construction.

P3: Collateral intervention. All four arms solve all 80 clean tasks, and the governed arms record no interventions on these runs. Performance is also largely preserved on the reroutable tasks: success is $20/20$, $19/20$, $20/20$, and $20/20$ for OFF, Fixed-7, Budget, and DWELL, respectively. Budget is the only governed arm to intervene on a reroutable task ($1/20$). These results indicate little collateral intervention on clean or successfully reroutable trajectories in this evaluation.

Provenance. Pre-registration `phase0/ARM640_PREREGISTRATION.md` (SHA-256 `0675aff2...b973`); deployment epoch `qwen38-27b-fp16-` (self-hosted vLLM, FP16 weights, native thinking, strict JSON), documented in `phase0/QWEN38_27B_FP16_2.md`; trajectory source commit `ae43fb2`; analysis commit `e003516`. Results, the per-trajectory table, paired tests, safety table, cost clocks, and the frozen report with its wording-boundary section are in `phase0/results_arm640_qwen38fp16/`. This matrix is a different model precision, serving stack, and epoch from the Qwen3.5-27B-Q8 rows and is never pooled with them.

C Additional Results

C.1 Failure-family coverage map

Figure 5 summarizes the three constructed failure families of Section 4 and the instrument that covers each.

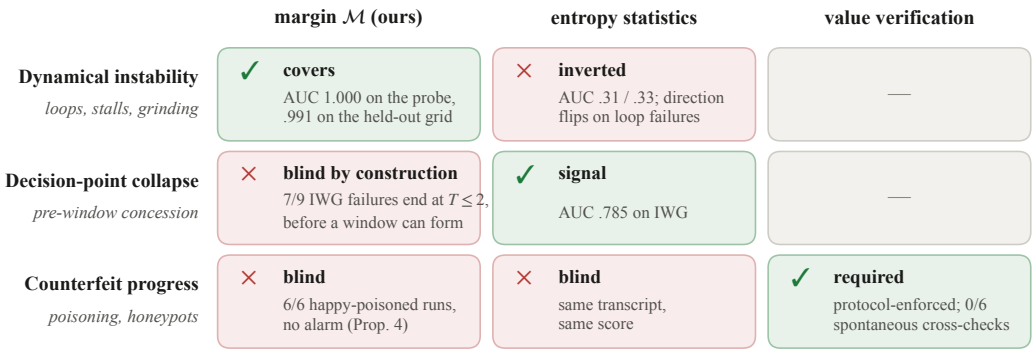


Figure 5: No single scalar diagnoses every failure. The margin specializes in dynamical instability; entropy remains useful for decision-point collapse; counterfeit progress requires value-level verification.

The results above establish that the runtime margin can detect dynamical stagnation, but they do not imply that every orchestration failure is observable from trajectory dynamics. We therefore examine where the margin provides a useful signal and where additional information is required. Figure 5 summarizes these coverage boundaries.

Dynamical instability. Loops, repeated retries, and prolonged stalls are the intended setting for the margin. In these cases, the trajectory continues long enough for changes in progress and execution burden to accumulate. The controlled probe and held-out evaluations in Section 4 provide the main evidence for this regime, with M^{doc} AUCs of 1.000 and 0.991, respectively.

Decision-point collapse. A different failure mode occurs when the run terminates before enough trajectory history is available to form the runtime window. On the external IWG set, seven of the nine failed runs end within the first two steps (five at $T = 1$ and two at $T = 2$), before the margin can become active. Accordingly, the margin detects only 11% of these failures, whereas an entropy-based statistic reaches AUC 0.785. Such early concessions therefore require signals available at the decision point rather than statistics that depend on a sufficiently long trajectory.

Counterfeit progress. The margin can also fail when the observed trajectory appears to make progress even though the underlying information is wrong. When a terminal fact is silently corrupted, all $6/6$ tested runs complete with incorrect answers and no margin alarm is raised; the runtime observer overestimates progress by 0.20. In contrast, when the corrupted value is checked downstream, the inconsistency becomes an observable stall and all $6/6$ runs trigger an alarm. When two sources contain one correct and one poisoned value, the model spontaneously cross-checks them in $0/6$ runs. These results show that trajectory dynamics can detect corruption only when it changes the subsequent execution trace; otherwise, explicit value verification is required.

Together, these cases define the scope of the runtime margin. It is effective for dynamical loss of progress, while very early termination and transcript-consistent false progress require complementary observation channels.

C.2 Trivial-monitor baselines

Independent stall-fraction check. As an additional validity check on the controlled probe, we compare $\mathcal{M}$ with a stall fraction defined without reference to the margin: the fraction of transitions on which progress z (the cumulative share of matched gold checkpoints) fails to strictly increase. We restrict the comparison to the 51 runs that satisfy the margin's eligibility gate of three valid transitions and obtain Pearson $r = 0.98$ over these trajectories. Since the stall fraction and $\mathcal{M}$ both depend on the same progress observation z, the correlation supports the claim that $\mathcal{M}$ tracks trajectory-level stagnation rather than that z is independently confirmed.

Table 6 compares the margin with two trivial monitors on the frozen trajectories already reported in Sections 3 and 4 (offline, no LLM reruns; Governor-arm trajectories excluded). B1 is a progress-stall rule that alarms after $w = 6$ steps without a new checkpoint; B2 is the three-consecutive-failure trigger of Governor v1. Both share the margin's eligibility gate (three valid transitions), so a run with $T \leq 2$ is ineligible for every rule. Three further observations are secondary and descriptive, not confirmatory claims. (a) Secondary: along the E2 ladder (clean, two, one, zero healthy holders) the mean M^{doc} decreases monotonically ($+1.645$, $+.995$, $+.751$, $+.022$), whereas the mean B1 stall length is $0/0/0/8.67$ and the mean B2 failure streak $0/0/0/4.0$; both trivial scores move only at the last rung. (b) Secondary: on the 18 collapsed E2 runs with zero healthy holders, the margin ($\tau = -.02$) detects 17, B1 8, and B2 10, with median first alarms at steps 3, 7, and 7. (c) Exploratory: on the 160 ungoverned (OFF) trajectories of the Qwen3.8 FP16 four-arm matrix, B1 and the margin at $\tau_{\mathrm{on}} = 0$ give identical alarm indicators on 160/160 runs. Source: `phase0/results_monitor_baselines_20260906/`.

Weighting ablation (post-hoc). Section 2 notes that the implemented executor grouping departs from consistent counting. Recomputing completed-trajectory AUCs on the full frozen sets from the same log-potential changes under three weightings—the implemented $\mathcal{M}$, M^{doc}, and the consistent-counting net drop rate—gives $.968/1.000/1.000$ on the probe ($n = 63$), $.989/.991/.987$ on the held-out grid (48), $.990/.990/.989$ on E1 (144), and 1.000 for all three on E2 (72) and E3 (35). The M^{doc} values reproduce the frozen reports; on the common eligible sets of Table 6, $\mathcal{M}$ and M^{doc} both reach 1.000 on every set. The executor grouping therefore adds no measurable separation on these data. Source: `RESULTS_AUDIT.md` (2026-09-26 entry).

Table 6: Margin versus two trivial monitors on frozen trajectories (offline, no LLM reruns). The margin column is M^{doc} in Q1 and the windowed $\mathcal{M}_k$ at the stated threshold in Q2–Q3. B1: progress-stall, alarm after $w = 6$ steps without a new checkpoint; B2: three consecutive failures on one executor (Governor v1 hard trigger); all rules share the margin eligibility gate (three valid transitions). Q1: AUC on the common set ($T \geq 3$ and at least one eligible step) of M^{doc}, the B1 stall length, and the B2 streak length; paired differences are task-level cluster bootstrap 95% CIs (10,000 resamples). Q2: the 18 collapsed E3 runs (leaf/terminal/merge, six each), margin at $\tau_{\mathrm{on}} = 0$, median first-alarm step; paired differences in steps over runs alarmed by both rules (negative: margin earlier; leaf Margin−B1: two runs). Q3: successful recoverable runs (E2 with one or two healthy holders, held-out reroute/wide/dead-end, original and Qwen3.8); counts with rates in parentheses, paired rate differences (positive: margin alarms more often); Clopper–Pearson intervals in the source. Source: `phase0/results_monitor_baselines_20260906/`.

Set	Margin	B1	B2	Margin−B1 [95% CI]	Margin−B2 [95% CI]
Q1: separation AUC on the common set					
Probe (51)	1.000	.907	1.000	+.093 [0, .300]	0 [0, 0]
Held-out (45)	1.000	.976	.979	+.024 [0, .057]	+.021 [0, .051]
Qwen3.8 held-out (48)	1.000	.936	.928	+.064 [.002, .143]	+.072 [.004, .155]
E1 (138)	1.000	.974	.973	+.026 [.007, .058]	+.027 [.007, .061]
E2 (71)	1.000	.950	.950	+.050 [0, .500]	+.050 [0, .500]
E3 (34)	1.000	1.000	1.000	0 [0, 0]	0 [0, 0]
Q2: E3 collapsed runs ($n = 18$), $\tau_{\mathrm{on}} = 0$					
Detected (of 18)	17	12	17		
First alarm, leaf	4	6.5	4	−4 [−4, −4]	0 [−.5, 0]
First alarm, terminal	6	7	4	−1 [−2, −1]	+2 [1, 2]
First alarm, merge	7	8	4	−1 [−1.5, −1]	+3 [2.5, 3]
First alarm, all strata	6	7	4	−1 [−4, −1]	+2 [0, 2.5]
Median window $T - k$	7	8.5	8		
Q3: alarms on recoverable successful runs					
Pooled (60), $\tau = -.02$	7 (.117)	6 (.100)	10 (.167)	+.017 [0, .060]	−.050 [−.125, 0]
Pooled (60), $\tau_{\mathrm{on}} = 0$	10 (.167)	6 (.100)	10 (.167)	+.067 [.013, .156]	0 [0, 0]
By set at $\tau = -.02$					
E2, one or two healthy (33)	0	0	0		
Held-out (11)	2	1	3		
Qwen3.8 held-out (16)	5	5	7		

Replay summary. The four-arm results establish the consequences of different stopping ceilings, but do not by themselves identify which runtime signal drives the intervention. We therefore perform a post-hoc replay analysis on all 1,419 governed trajectories from the primary Qwen3.8-27B epoch and the two replication epochs, Qwen3.6-27B and Qwen3.5-9B. When we remove the runtime margin $\hat{\mathcal{M}}_k$ and keep only the three-failure trigger, the intervention time does not change on any faulted trajectory. The margin fires strictly earlier than the three-failure rule on 0% of faulted runs for Qwen3.8-27B and Qwen3.6-27B, and on at most 1.64% for the 9B model. Thus, on these workloads, intervention timing is driven primarily by the failure counter and the DWELL-and-budget arithmetic in (5). The margin $\hat{\mathcal{M}}_k$ instead serves as the trajectory-level measurement studied in Sections 3 and 4. The replay is described next.

In-loop redundancy of the margin channel (post-hoc replay). The natural in-loop monitor swap—fixed Governor, margin replaced by the B1 stall rule or removed entirely—can be screened on frozen trajectories, because the Governor's tape records the per-tick quantities both signals are computed from ($\widehat{M}_k$, $\hat{z}$, failure streaks, regime). We replayed the deployed alarm state machine on all 1,419 governed trajectories of the three FP16 epochs under three alarm rules (deployed margin, B1 on $\hat{z}$, hard-trigger only), sharing the failure trigger, the escalation ladder, and the logged regime sequence; the deployed rule's replay reproduces the taped `alarm_run`/`level` series exactly on every trajectory. On faulted runs the margin almost never leads the hard failure trigger (strictly earlier in 0/240 (3.8), 0/225 (3.6), and 3/240 (9B) runs; a lead of two or more plan cycles in at most 1.6%), and removing it leaves every faulted-run L2/L3 escalation time unchanged. Table 7 separates these results from the two L2 timing changes under B1 replacement and the clean-run final orders. Its only distinct in-loop footprint is a small clean-run false-alarm channel (margin-only alarms on 2.5–6.25%

of 9B clean runs), which fully accounts for the Budget arm's two clean final orders of Appendix C.13. On this workload the in-loop trigger is therefore carried by the failure streak and the dwell arithmetic; the margin's contribution is the offline coverage of Sections 3–4, not intervention timing. Replay validity is exact up to the first tick at which a swapped monitor would have changed an intervention. Source: `phase0/ARM640_MONITOR_SWAP_STAGE0_20260919.md`.

Table 7: Trigger attribution from fixed-tape state-machine replay (post-hoc). The deployed rule reproduces the logged `alarm_run/level` series on all 1,419 valid governed runs (480/459/480 for 3.8/3.6/9B), with zero mismatches. H removes the margin channel and retains the hard failure trigger; $\widehat{B}_1$ replaces the margin with the runtime progress-stall rule. The failure trigger, escalation ladder, and logged dwell-regime sequence are held fixed. Counts concern runs, not individual alarms or orders.

(a) Alarm lead and escalation timing

Epoch	Margin strictly before H (faulted)	H changes L2/L3 timing (faulted)	$\widehat{B}_1$ delays L2 (all governed)
3.8	0/240	0/240	1/480
3.6	0/225	0/225	1/459
9B	3/240	0/240	0/480

(b) Margin-only alarms and final orders on 9B clean runs

Arm	Margin-only alarm	Logged final order	Logged order absent under H
Fixed-7	3/80 (3.75%)	0/80	0/80
Budget	5/80 (6.25%)	2/80	2/80
Full	2/80 (2.50%)	0/80	0/80

The three strictly earlier faulted alarms are in 9B Fixed-7 (two) and Budget (one); they do not change L2/L3 timing under H replay. The two $\widehat{B}_1$ timing exceptions are in 3.8 Budget and 3.6 Full; a state-timing change need not change an emitted intervention. The two logged 9B clean final orders are absent under H on the recorded input tape. A replay comparison is exact only up to its first changed intervention; it does not establish subsequent model behavior, task success, or closed-loop causal effects. Source: the frozen replay summary recorded in `RESULTS_AUDIT.md` (2026-09-19).

C.3 Governor mechanism audit

The six trajectories in the mechanism audit separate two cost clocks: each has seven tool-bearing planner cycles but nine executor calls, because two healthy-branch calls run alongside the first two failures. Wrap-up and L3 coincide after the seventh target failure, and the obeyed final order adds no tool step (Proposition 2).

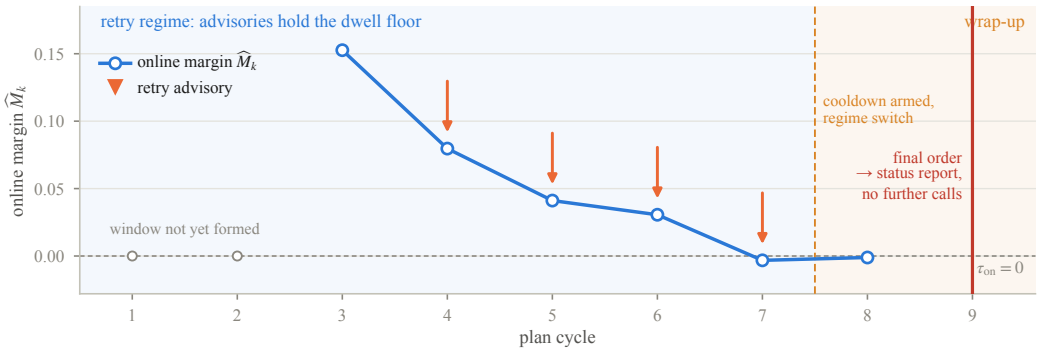


Figure 6: A governed unsolvable run. Retry advisories preserve dwell while the posterior boundary permits it; after the regime switch, a final order elicits an honest status report instead of exhausting the budget.

C.4 Intervention and deployment attribution

Table 8 is the primary Qwen3.6-27B three-arm evaluation discussed in Section 5.

Table 8: Primary Qwen3.6-27B evaluation of OFF, fixed-threshold v1, and adaptive DWELL (v2). †: v1 vs. OFF, $p = .041$; ‡: DWELL vs. OFF, $p = .33$; ⋆: one-sided stop-length test, $p = 4 \times 10^{-4}$. Source: `phase0/results_gov/`.

Terrain	Metric	OFF	v1	DWELL
Fragile	success	.69	.33†	.56‡
	mean steps	7.1	3.1	5.8
Unsolvable	median steps	13	3	7⋆
Reroutable	success	.83	1.00	1.00
	intervention	–	0	0
Clean	success	1.00	1.00	1.00
	intervention	–	0	0
Faulted	steps / success	19.2	–	16.6

Table 9: Cross-model Governor results: success OFF/v1/v2; stop = median unsolvable length. †: multiple deployment epochs; ‡: faulted-only; §: four-arm FP16 entries OFF/Fixed-7/Full (Budget omitted), with Fixed-7 replacing the distinct v1 rule. Sources: Appendix B.

Model / deployment	n	Fragile success	Unsolvable stop	Clean OFF/v2
Qwen3.6-27B	240	.69/.33/.56	13/3/7	1.00/1.00
DeepSeek V4 Flash†	84	1.00/.33/.50	24/3/7	1.00/1.00
DeepSeek V4 Pro	84	1.00/.33/.50	25/3/7	1.00/1.00
GLM-5.2	84	.58/.33/.25	15/3.5/7	1.00/.83
Qwen3.8-27B ‡	72	.92/.33/.50	12/3/7	–/–
Qwen3.8-27B FP16 (self-hosted)§	640	1.00/.63/.53	25/8/7	1.00/1.00
Qwen3.5-9B FP16 (self-hosted)§	640	1.00/.63/.53	25/8/7	1.00/1.00

Three additional orchestrators receive the same 84-run matrix: 24 faulted OFF, v1, and v2 trajectories, plus six clean OFF and six clean v2 controls. Qwen3.8-27B receives the same frozen 24 faulted specifications in each arm (72 runs total), with no Governor clean controls.

Adaptive v2 reduces the unsolvable median to seven steps in all five evaluations. The one-sided tests are significant for Qwen3.6 and both DeepSeek variants; the six-seed GLM comparison has the same direction but is underpowered ($p = .187$). On Qwen3.8, v1 changes the unsolvable median $12 \to 3$ ($p = .0048$), while v2 changes it $12 \to 7$ ($p = .1736$). Fragile success on Qwen3.8 changes $11/12 \to 4/12 \to 6/12$ for OFF/v1/v2; the v1 and v2 Fisher tests against OFF give $p = .0094$ and $p = .0686$. The frozen cross-model conclusion is: stop-loss transfers; success-preservation safety does not.

Actual intervention logs are essential for attribution. Reroutable runs have zero v1/v2 intervention in all three 84-run replications, and all corresponding clean v2 runs also have zero intervention. In Qwen3.8, reroutable success is $6/6$ in every arm with zero Governor intervention; v1 and v2 each intervene on $14/24$ faulted runs. v1 records 14 advisories; v2 records 56 advisories, 12 cooldowns, 12 final orders, and zero vetoes. No Governor clean task was run for Qwen3.8, so the Holdout's clean alarms cannot substitute for a clean intervention control. These Qwen3.8 counts come from `phase0/results_gov_qwen38_27b/`. Consequently, the GLM clean difference ($6/6$ vs. $5/6$) and reroutable variation are trajectory stochasticity, not observed Governor actions. The Flash matrix additionally crosses provider epochs. A separate DeepSeek V4 Pro deployment audit finds stable outcome categories but large resource differences (Appendix C.9), so we do not pool providers or treat a model alias as a deployment identity.

The K3 OFF/v1/v2 batch remains excluded: its formal run was gated after an external quota failure. We retain that incomplete batch rather than mixing it with the completed five-model evidence.

C.5 The kappa frontier

Figure 7 visualizes the frozen patience–stop-loss frontier discussed in Section 5.

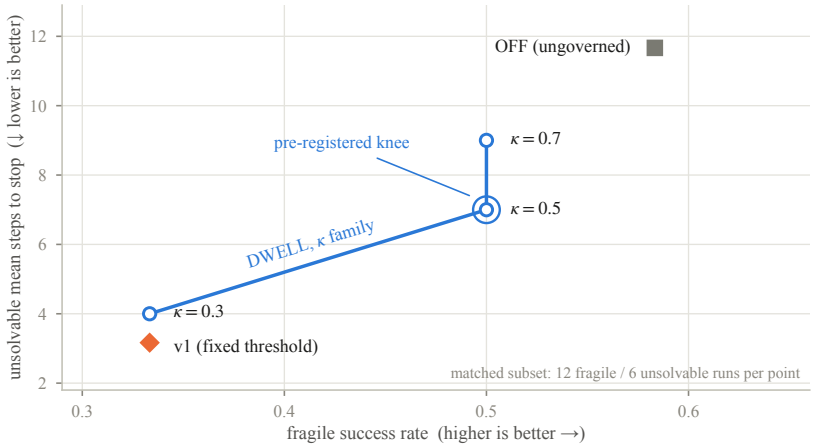


Figure 7: Patience–stop-loss frontier. Increasing κ increases allowed dwell. The frozen $\kappa = .5$ lies at the observed knee; $\kappa = .7$ pays additional unsolvable steps without improving the measured fragile success rate.

C.6 Exploratory: lost-pair counterfactual and throughput breakeven

Everything in this subsection is exploratory. It was not pre-registered, it uses only the frozen four-arm trajectories, and it changes none of the P1–P3 conclusions of Section 5.2. Source: `phase0/results_arm640_qwen38fp16_20260902/exploratory/`.

Fault realizations are identical across arms (exploratory precondition). In the generator, the outcome of the j-th post-onset call to the faulted holder is determined by (seed, trial, j) alone (`stressgen.py`, lines 236 and 246–248), and vetoes do not consume the random stream. Replaying the seeded stream against all 320 faulted trajectories reproduces every one of the 1,757 sampled outcomes (0 mismatches, 0 vetoes). The fault side of the counterfactuals below is therefore exact; the planner side is not, because governed arms receive advisory text that OFF never sees.

Lost pairs are threshold sets of k_{win} (exploratory). Let k_{win} be the longest failure streak an OFF fragile trajectory survives before the target call that succeeds. All $40/40$ OFF fragile runs succeed; k_{win} has median 4.5 and maximum 19, with $P(k_{\text{win}} \geq 3) = .700$, $P(k_{\text{win}} \geq 7) = .375$, and $P(k_{\text{win}} \geq 9) = .300$. Task by task, the lost pairs of each governed arm are exactly the tasks whose k_{win} reaches that arm's effective wrap-up threshold: Fixed-7 loses the 15 tasks with $k_{\text{win}} \geq 7$; Full, whose threshold from Equation (6) is $7/6/4$ for $c_0 = 2/5/10$, loses $2 + 7 + 10 = 19$; Budget, whose effective threshold is $7/4/3$, loses $2 + 9 + 12 = 23$. In all 57 lost pairs the planner made zero target calls after the first wrap-up intervention and obeyed every final order ($20/20$; forced stops 0): on this model a wrap-up advisory is operationally a stop. The exploratory reading of the reversed P2 direction is that the Full rule treats budget already spent on healthy work (large c_0) as a reason for less patience, so its threshold is lowest on the $c_0 = 10$ stratum, where it loses 10 of its 19 pairs, whereas Fixed-7 applies the same patience everywhere. Figure 8 displays these observed threshold sets and marks the two over-budget OFF successes separately.

κ counterfactual (exploratory, one-sided). For each of Full's 19 lost pairs we compute κ_{needed}, the smallest κ at which Full would not have entered wrap-up at any point along OFF's realized failure streak. This is a one-sided counterfactual—if Full did not wrap up and the planner continued exactly as OFF did along the identical fault realization—and not a re-run. Seventeen values are finite (median 1.833, quartiles $.692/3.400$, maximum 21.0); two are infinite because OFF needed 28 and 30 executor calls, beyond $B = 25$ (implementation note: the Governor clocks the plan-cycle budget against executor calls, so no κ recovers these two). On the grid $\kappa \in \{.5, .7, 1, 1.5, 2, \infty\}$, the number of lost pairs whose streak Full would have outlasted is $0/6/7/8/11/17$ of 19, while the wrap-up point on a standard unsolvable task ($c_0 = 2$) moves to $k_{\text{wrap}} = 7/9/11/14/15/23$ consecutive failures. No κ on this grid retains fragile success without postponing the unsolvable stop by a comparable

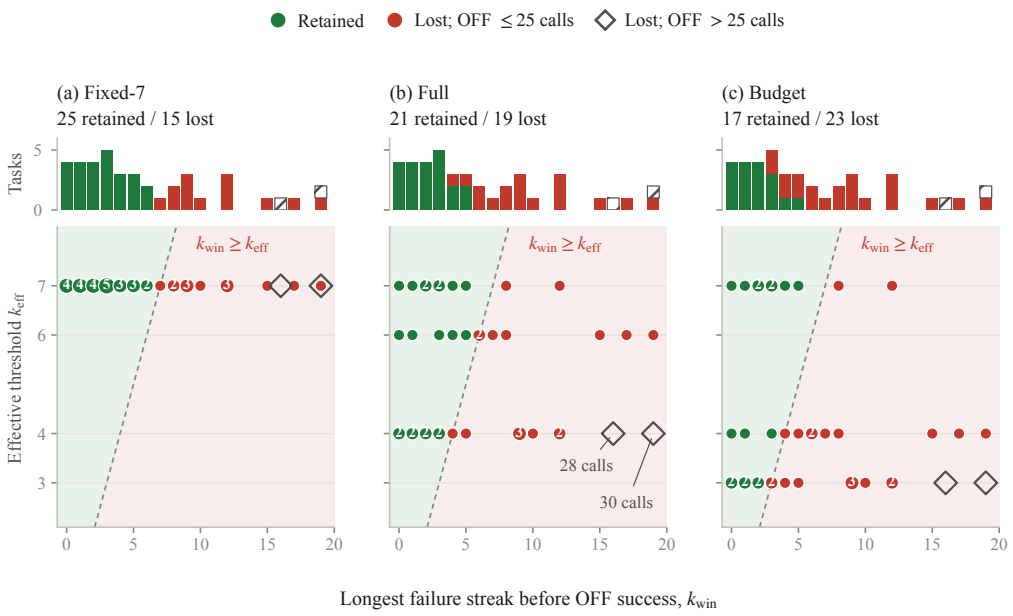


Figure 8: Success loss and effective stopping thresholds on the 40 primary Qwen3.8 fragile tasks (exploratory). The horizontal axis is OFF's longest failure streak before success, k_{win}; the vertical axis is the effective intervention threshold, k_{eff}. Filled circles aggregate tasks at the same coordinates; internal numbers denote multiplicity when larger than one. Histograms show task counts. Hollow diamonds and hatched bars identify t35/t36, whose OFF successes require 28/30 calls; in Fixed-7, t36 overlaps a within-budget lost task. The headings retain the original 15/19/23 lost-success counts. Under 25-call scoring these become 13/17/21; Full's 19-to-17 change matches Table 10. The observed losses satisfy $k_{\text{win}} \geq k_{\text{eff}}$ on this sample. This relation does not establish outcomes under a different intervention, and the effective threshold need not equal the first wrap-up regime switch.

number of failures; the two quantities move together along one frontier. This is not a statement about what Full's success rate would be at another κ; that would require a new pre-registered run.

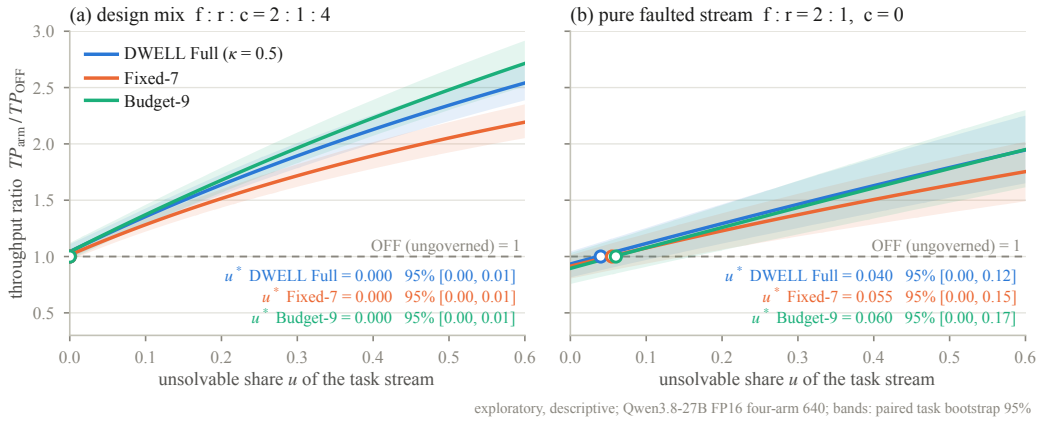


Figure 9: Exploratory throughput breakeven on the four-arm matrix. Successes per million tokens of each governed arm relative to OFF, as a function of the unsolvable share u of the task stream. Left: the remaining tasks keep the design ratio fragile:reroutable:clean = 2:1:4; right: no clean tasks. Bands are 10,000-sample task-level paired bootstrap 95% intervals; markers show the fine-grid breakeven u^*. Descriptive only; u^* depends on the assumed mix and is not a constant.

Throughput breakeven (exploratory). Figure 9 scans the unsolvable share u. At the design ratio, the ratio of governed to OFF successes per million tokens is at or above one already at $u = 0$ on

the grid (1.014 [.965, 1.064], 1.042 [.965, 1.121], and 1.043 [.980, 1.109] for Fixed-7, Budget, and Full), but every $u = 0$ interval includes one, so that point supports no direction; from $u \geq .05$ the bootstrap lower bounds exceed one. With no clean tasks (fragile:reroutable = 2:1) the $u = 0$ ratios are .915/.895/.934 and the grid breakeven is $u^* = .10/.10/.05$ (fine grid .055 [0, .155], .060 [0, .165], .040 [0, .125]). Under the user's lens, each lost fragile success saves 27,638/28,303/29,865 tokens in the fragile stream, against the 39,551 tokens OFF spends per fragile success, whereas each unsolvable task saves 73,941/85,035/81,344 of OFF's 111,489 tokens. Whether governance pays is therefore a value judgement between operator budget and user harm, and it turns on u; this is why a runtime must know the fragility gap before it chooses a ceiling.

C.7 Budget-matched harm and gap sensitivity

Everything in this subsection is a post-hoc sensitivity on frozen trajectories; no new runs, and the pre-registered endpoints are unchanged. The OFF arm has no call budget (Appendix B.1), so the harm screen and the fragility gap compare a governed arm against an opponent that can spend beyond the Governor's $B = 25$ call budget. We rescore every OFF fragile/faulted success with $N_{\text{call}} > 25$ as not achievable within budget, and recompute both quantities under this hard cap. Only contrasts involving OFF are affected: Full, Fixed-7, and Budget share the same budget arithmetic, and the asymmetry inflates only the magnitude of P1's stop-loss, never its direction.

Table 10: Common-budget comparison on frozen trajectories (post-hoc sensitivity, not a new closed-loop experiment). All arms share the external $T \leq 25$ plan-cycle cap; the three governed arms also use $B = 25$ executor calls, while OFF has no call cap. Here an OFF success counts under the common budget only if $N_{\text{call}} \leq 25$; Full is unchanged. (a) Success counts and Full minus OFF success-rate differences on valid pairs; brackets are paired-bootstrap 95% CIs (10,000 resamples, seed 20260901), with two-sided exact McNemar p below. (b) Gap uses all valid OFF fragile/faulted runs, a different denominator in 3.6, and the same frozen $\text{mean}(\pi^{25-c_0})$ baseline.

(a) Fragile/faulted success under a common call budget

Epoch	Pairs	Full	OFF	Full − OFF success rate	
	n		observed → capped	Observed	$B = 25$ scoring
3.8	40	21	40 → 38	$-.475$ [$-.625, -.325$] $p = 3.81 \times 10^{-6}$	$-.425$ [$-.575, -.275$] $p = 1.53 \times 10^{-5}$
3.6	34	18	31 → 30	$-.382$ [$-.559, -.235$] $p = 2.44 \times 10^{-4}$	$-.353$ [$-.529, -.206$] $p = 4.88 \times 10^{-4}$
9B	40	21	40 → 38	$-.475$ [$-.625, -.325$] $p = 3.81 \times 10^{-6}$	$-.425$ [$-.575, -.275$] $p = 1.53 \times 10^{-5}$

(b) OFF over-cap successes and fragility-gap sensitivity

Epoch	OFF n	Over-cap successes (calls)	Non-successes: obs. → capped	Gap: observed → capped
3.8	40	2 (28, 30)	0/40 → 2/40	$-.086 \to -.036$
3.6	36	1 (27)	3/36 → 4/36	$+.0003 \to +.028$
9B	40	2 (27, 30)	0/40 → 2/40	$-.086 \to -.036$

Two tasks account for every over-cap success (t35 and t36; 3.6 has t35 only): the same tasks that Appendix C.6 identifies as beyond any finite κ. Under the matched budget the harm point estimates shrink by .03–.05 and all three screens remain significant; the hard-cap gap moves toward zero by the same two collapses. The qualitative conclusions are unchanged, and the matched numbers, not the pre-registered ones, are the conservative reading of the harm screen. The transfer epoch of Appendix C.14 realizes this matched design prospectively: all four arms, including OFF-B, run under one harness call cap, so no rescoring is needed there. Source: `phase0/results_arm640_budget_matched_20260919/`; read-only entry point `phase0/arm640_budget_matched.py`.

C.8 K3 dynamic observation

On the 36-run K3 dynamic set, each of four conditions has nine independent tasks. At $w = 6, \tau = -.02$, gold/runtime persistent-fault recall is $8/9$ and $4/9$ respectively; clean false alarms and recoverable-fallback alarms are both $0/9$ for both observers. Median $\widehat{z}$–z correlation is 1.000, mean absolute progress error is .009, median $\widehat{V}$–V correlation is .981, and alarm agreement is .889.

Raising the threshold restores runtime recall but would constitute post-hoc K3 calibration, so the main result retains the frozen operating point.

C.9 Cross-deployment sensitivity

DeepSeek V4 Pro was observed through a complete 84-run Aliyun matrix and a 10-run partial matrix on the official API. The deployments share identical task specifications but are not pooled. Across eight overlapping OFF/faulted tasks, answer outcome and termination category agree in every pair; four trajectory lengths differ by one or two steps. For the shared `pi100_s0` smoke task, both deployments produce the same arm-level termination pattern: OFF reaches 25 steps, v1 stops at three, and v2 stops at seven.

Resource trajectories are not invariant. On the shared OFF smoke task, the official deployment uses 55 rather than 78 API calls (-29.5%), 149,164 rather than 247,565 tokens (-39.7%), and 1,239.935 rather than 2,165.854 seconds of accumulated API latency (-42.7%). This is descriptive evidence, not a deployment-equivalence test: current outcomes appear robust on the overlap, while calls, tokens, latency, and fine-grained trajectories remain provider dependent.

C.10 Reproducibility checklist

- Every task world is regenerated from a stored seed and specification.
- Clean/faulted comparisons share topology, values, model, and budget.
- The parser, potential weights, window, threshold, and κ are frozen before held-out and K3 analyses.
- Raw trajectories contain model identity, condition, trial, planner process, entropy, executor traffic, and Governor events.
- Provider changes define new deployment epochs; cross-provider trajectories are paired when possible and are never silently pooled.
- Partial and zero-step runs are retained in explicit invalid directories and excluded from inferential tables.

C.11 Same-design Qwen3.6-27B FP16 comparison

Cross-model summary. The two replication epochs also allow us to test whether the outcome-level trade-off observed on Qwen3.8-27B transfers to other deployments. On Qwen3.6-27B, DWELL stops a paired median of 18 plan cycles earlier than OFF, while fragile-task success is $18/34$ for DWELL and $31/34$ for OFF. On Qwen3.5-9B, the complete $640/640$ matrix shows the same pattern: DWELL stops a paired median of 18 plan cycles earlier than OFF, but retains $21/40$ fragile successes compared with OFF's $40/40$. Thus, the stop-loss effect transfers across these model settings, together with its cost in recoverable success. The Qwen3.6-27B epoch contains arm-dependent invalid trajectories, and the corresponding attrition and sensitivity analyses are reported in Appendix C.11; complete Qwen3.5-9B results appear in Appendix C.13.

Role, deployment, and validity. The separately frozen addendum `phase0/ARM640_QWEN36_COMPARISON_ADDENDUM_20260907.md` specifies a second comparison epoch on the same 80 tasks, four arms, seeds, hardware, precision, and frozen controller parameters. Qwen3.8 FP16 remains the primary four-arm evaluation. For Qwen3.6 epoch-b, the serving template was aligned to deliver the same low-effort instruction; the deployment amendment records partial compliance with that instruction. The matrix completed all 640 cells, yielding 612 contract-valid trajectories and 28 contract-invalid cells, with nine additional records from interrupted blocks archived separately. The completed-data analysis retained the original validity rules, endpoints, directions, and multiplicity families; permission to analyze the reduced pairs was recorded after the attrition audit. Each contrast uses its own available paired tasks. Inference from these pairs to the complete planned task set depends on the missingness mechanism.

Stop-loss and success. All six pre-specified P1 tests have 13 available pairs, all nonzero and in the specified direction (exact one-sided Wilcoxon, raw $p = 2^{-13}$, six-test Holm $p = .0007324$

each). Full reduces N_{call} and T relative to OFF by paired medians of 18 $[16, 20]$ on both clocks; Fixed-7 minus Full is 3 $[1, 3]$, and Full minus Budget is 1 $[1, 2]$. The OFF contrasts omit tasks 40,41,44,47,54,57,58; the other contrasts omit only 47,54. The valid Fixed-7, Budget, and Full cells for tasks 57 and 58 remain in their respective comparisons.

Table 11: Qwen3.6 FP16 P2 results on available valid pairs. Differences and 95% paired bootstrap intervals are Full minus comparator. The OFF row is the separately specified two-sided harm screen.

Comparator	Pairs	Full succ.	Other succ.	p_{Holm}	Difference [95% CI]
Fixed-7	35	19	22	1.000	$-.0857$ $[-.2000, .0000]$
Budget	37	18	15	.375	$+.0811$ $[-.0270, .1892]$
OFF	34	18	31	—	$-.3824$ $[-.5588, -.2353]$

Both P2 superiority tests remain below their required evidence threshold. The discordant counts (Full-only/comparator-only success) are $0/3$ for Fixed-7 and $4/1$ for Budget. The OFF harm screen has $0/13$ discordances and two-sided exact $p = .0002441$; a budget-matched sensitivity for this screen appears in Appendix C.7. Missing tasks are 1,5,21,37,39 for Fixed-7; 1,25,39 for Budget; and 1,8,24,30,37,39 for OFF.

Collateral intervention and baseline persistence. Clean success/valid counts are $79/79$, $78/78$, $78/78$, $78/78$ for OFF, Fixed-7, Budget, Full; reroutable/faulted counts are $20/20$, $20/20$, $19/19$, $17/17$. Every listed group has zero recorded interventions. For the governed clean groups the rule-of-three upper bound is $3/78 = .0385$; for governed reroutable groups it is .150, .158, .176, respectively. OFF solves $33/36$ valid fragile/faulted tasks: collapse $3/36 = .08333$ (95% Clopper–Pearson $[.01753, .22469]$), against the frozen $\text{mean}(\pi^{25-c_0}) = .08305$ over those same 36 tasks. The observed gap is .00029. This epoch adds a second near-zero-gap, high-persistence setting; the earlier Qwen3.6 three-arm deployment remains a distinct evaluation.

Attrition and recovered-outcome sensitivity (exploratory). Faulted invalid counts over 80 cells per arm are OFF 6, Fixed-7 3, Budget 2, Full 10; clean counts are 1,2,2,2. All 28 excluded cells contain a total of 31 length-limited, empty-content attempts, with failed-attempt reasoning lengths 31,132–32,103 tokens. Every affected logical call subsequently recovers, and all final task outcomes and usage records are available; the original contract-invalid labels remain unchanged. A separate recovered-outcome sensitivity calculation retains these final outcomes: Full's paired median T saving versus OFF is 17 $[14.5, 18]$ on 20 tasks; Full/Fixed-7/Budget/OFF fragile successes are $20/25/17/36$ out of 40. The two P2 superiority p_{Holm} values remain 1 and .375.

Independently of this recovery interpretation, binary missing-outcome bounds retain observed valid-arm outcomes and let each missing arm range over $\{0, 1\}$. Full minus Fixed-7 on all 40 tasks lies in $[-.175, -.050]$; Full minus Budget in $[.050, .125]$; Full minus OFF in $[-.450, -.300]$. These are identification bounds, with sampling uncertainty considered separately. The arm-dependent attrition motivates both audits; its causal mechanism and selection-bias direction remain open in this dataset.

Source and reproduction. Source [Q36F-ARM]: `phase0/results_arm640_qwen36fp16b_20260913/`. The read-only entry point `phase0/analyze_qwen36_completed.py` reuses the frozen pairing, parser, and statistical functions and emits per-trajectory, paired-value, cost, validity, and sensitivity tables. The execution note is `phase0/QWEN36_ANALYSIS_EXECUTION_NOTE_20260913.md`. The formal 640-cell bundle digest is `34afa35e...f531`; the completed driver-state digest is `e4a0059b...5698`. Each model epoch retains separate estimates and within-epoch intervals; all cross-model comparisons are descriptive.

C.12 Fixed-workload token efficiency (exploratory)

Success per token (moved from the main text). We finally examine the trade-off in terms of token usage. For each model, we compare the number of successful task completions per token under DWELL and OFF. A ratio above 1 means that DWELL achieves more successes for the same token cost. The DWELL/OFF ratios are 1.378 $[1.188, 1.563]$ for Qwen3.8-27B, 1.218 $[0.946, 1.525]$ for Qwen3.6-27B, and 1.387 $[1.186, 1.591]$ for the Qwen3.5-9B model. Thus, DWELL has higher

token efficiency than OFF by point estimate in all three epochs, although the advantage is less clear for Qwen3.6 and is not consistent relative to the simpler fixed stopping rules. These results are exploratory and workload-specific; the accounting follows below and in Figure 10.

This completed-data analysis supplements the original endpoints. It fixes the faulted workload at 40 fragile, 20 unsolvable, and 20 reroutable tasks. For arm a, let Y_{ia} denote recorded task success, A_{ia} frozen contract validity, and K_{ia} the tokens consumed by all attempts in the formal cell. Contract-valid delivery efficiency is

$$\mathrm{Eff}_a = 10^6 \frac{\sum_{i=1}^{80} A_{ia} Y_{ia}}{\sum_{i=1}^{80} K_{ia}}. \tag{9}$$

Invalid cells contribute their actual token cost and zero contract-valid deliveries. This operational composite is distinct from the valid-pair P2 success endpoint. Tokens per delivery are its reciprocal, rescaled by 10^6. Within each epoch, 10,000 paired task bootstraps (seed 20260901) resample 40/20/20 tasks within their respective terrain strata and retain all arm observations for each sampled task.

Table 12: Fixed faulted workload, exploratory. All 80 formal cells per arm contribute token cost, including failed attempts. Valid successes satisfy both the original trajectory contract and the outcome parser.

Epoch	Arm	Valid/80	Valid succ.	Total tokens	Tokens/succ.
3.8 FP16	OFF	80	60	4,356,862	72,614
	Fixed-7	80	44	2,456,607	55,832
	Budget	80	37	1,995,242	53,925
	Full	80	41	2,159,735	52,676
3.6 FP16	OFF	74	53	6,840,651	129,069
	Fixed-7	77	43	4,014,935	93,371
	Budget	78	35	3,172,022	90,629
	Full	70	36	3,814,861	105,968
9B FP16	OFF	80	60	8,311,870	138,531
	Fixed-7	80	44	4,524,506	102,830
	Budget	80	36	3,717,151	103,254
	Full	80	41	4,094,447	99,865

Table 13: Full/comparator efficiency ratios and within-epoch 95% paired, terrain-stratified bootstrap intervals. Ratios above one favor Full. All quantities are exploratory and workload-specific.

Epoch	Full/OFF	Full/Fixed-7	Full/Budget
3.8 FP16	1.378 [1.188, 1.563]	1.060 [.954, 1.175]	1.024 [.934, 1.147]
3.6 FP16	1.218 [.946, 1.525]	.881 [.718, 1.049]	.855 [.674, 1.077]
9B FP16	1.387 [1.186, 1.591]	1.030 [.905, 1.191]	1.034 [.905, 1.206]

On the primary 3.8 workload, Full reduces tokens per valid success by 27.5% relative to OFF. Its point estimate leads the fixed rules, with both comparison intervals spanning one. On 3.6 the Full/OFF efficiency point ratio is 1.218, and the fixed rules have higher efficiency point estimates. On 9B the Full/OFF ratio is 1.387 [1.186, 1.591], again excluding one, while both fixed-rule intervals span one. These epoch-specific patterns separate evidence for resource savings from the additional value of the adaptive trigger.

Outcome and cost sensitivities. Using all recorded final successes after recovery gives 3.6 success counts 56/45/37/40 for OFF/Fixed-7/Budget/Full, with the same formal-cell costs. Full/OFF becomes 1.281 [1.060, 1.522], Full/Fixed-7 .936 [.808, 1.062], and Full/Budget .899 [.770, 1.045]. Separately adding the archived interrupted-block faulted token costs (OFF 143,560; Fixed-7 128,561; Budget 48,270; Full zero) to the contract-valid delivery calculation gives ratios 1.244 [.963, 1.564], .909 [.736, 1.097], and .868 [.681, 1.092], respectively. These sensitivities retain 80 formal cells per arm. Online throughput and hardware costs require separate measurements.

Utility frontier (post-hoc). Table 12 also reads as one utility problem, $U_a(v) = vS_a - C_a$ with v the token value of one incremental contract-valid success. An arm lies on the utility frontier when no cheaper arm delivers at least as much and the marginal cost per incremental success strictly increases along the frontier. Table 14 reports the flip points between adjacent frontier arms. On the primary 3.8 workload all four arms lie on the frontier; the 9B epoch preserves the same four-arm frontier with the boundaries shifted upward. On 3.6 Full leaves the observed frontier, which then runs Budget→Fixed-7→OFF—the same ordering as that epoch's efficiency ratios. Under the frozen paired bootstrap of Table 13 (the same 10,000 draws, seed 20260901), Full appears on the resampled frontier in 81.0% of draws on 3.8 (70.8% on 9B, 6.3% on 3.6) and Fixed-7 in 56.1% (66.3%, 94.1%), while Budget and OFF are essentially always present ($\geq 94.4\%$ and $\geq 99.9\%$). The flip-point intervals are wide and mutually overlapping, so the boundaries are decision aids, not rankings. This reading is not pre-registered, assumes a linear v, inherits the fixed 40/20/20 workload, and keeps the original budget basis: OFF carries no call cap in these epochs, so its boundary does not transfer to a capped deployment. The unified-cap analogue is the transfer epoch's E4 accounting (Appendix C.14), where capped OFF-B becomes utility-optimal only above ≈193,500 tokens per incremental success.

Table 14: Utility-frontier flip points between adjacent frontier arms on the fixed faulted workload (post-hoc, not pre-registered). $v^* = \Delta C/\Delta S$ is the token value of one incremental contract-valid success at which the utility-optimal arm switches. Intervals are paired, terrain-stratified bootstrap 95% percentiles on the frozen draws of Table 13, conditional on a positive success difference (last column). A negative interval endpoint marks resamples in which the nominally costlier arm was in fact cheaper. OFF has no call cap in these epochs; the unified-cap analogue is Table 17.

Epoch	Boundary	Δ tokens	Δ succ.	v^* [95% CI]	$P(\Delta S \geq 1)$
3.8 FP16	Budget→Full	164,493	4	41,123 [17,312, 179,441]	.985
	Full→Fixed-7	296,872	3	98,957 [37,423, 345,151]	.887
	Fixed-7→OFF	1,900,255	16	118,766 [89,361, 179,947]	1.000
9B FP16	Budget→Full	377,296	5	75,459 [−24,811, 300,923]	.995
	Full→Fixed-7	430,059	3	143,353 [23,821, 632,489]	.883
	Fixed-7→OFF	3,787,364	16	236,710 [169,269, 369,414]	1.000
3.6 FP16	Budget→Fixed-7	842,913	8	105,364 [51,322, 296,334]	>.999
	Fixed-7→OFF	2,825,716	10	282,572 [173,728, 707,857]	>.999

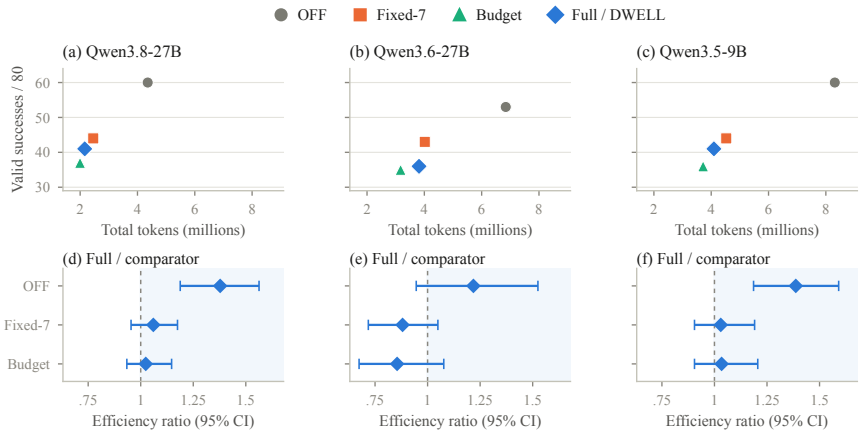


Figure 10: Cost and contract-valid delivery on the frozen faulted workload (exploratory; 40 fragile, 20 unsolvable, 20 reroutable tasks per arm). Top: recorded tokens and valid successes. Bottom: Full/comparator successes-per-token ratios with within-epoch, paired, terrain-stratified bootstrap 95% CIs (dashed line: one); every Full/fixed-rule interval includes one. Invalid cells contribute cost and zero deliveries. OFF has no call cap: original-run outcomes, distinct from the 25-call rescoring in Table 10. Values: Tables 12–13.

C.13 SAME-DESIGN QWEN3.5-9B FP16 COMPARISON

Role, deployment, and validity. The separately frozen addendum `phase0/ARM640_QWEN35_9B_ADDENDUM_20260918.md` specifies a third comparison epoch on the same 80 tasks, four arms, seeds, host machine, non-quantized weights, and frozen controller parameters, at roughly one-third of the primary model's scale. Qwen3.8 FP16 remains the primary four-arm evaluation. The 9B deployment serves bf16 weights with tensor parallelism two on two GPUs of the same host, a 32k context, and a chat template byte-identical to the 3.6 epoch's; the gate battery and its thresholds are copied verbatim from the frozen epochs. The formal matrix ran as eight serial driver shards over `block_index` mod 8 with byte-identical driver code, merged offline before analysis; the sharding amendment (`phase0/ARM640_QWEN359B_SHARD_GUARD_AMENDMENT_20260918.md`) leaves full-matrix behavior unchanged. Continuous batching can reorder reductions, so this epoch forms its own statistical family and is never pooled with the serial epochs. The matrix completed all 640 cells with 640 contract-valid trajectories, zero contract-invalid cells, and zero breaks—the first complete epoch of this design.

Gap probe and ignition. The addendum's Stage 1 is a gate-level OFF-arm probe over the 40 fragile/faulted tasks. It completed $40/40$ valid cells and measured a gap point estimate of $-.011$, below the pre-frozen .05 Stage-2 threshold. Under the addendum's sign-agnostic reporting rule, the full matrix was then fired by explicit override to obtain a powered boundary point (`phase0/ARM640_QWEN359B_OVERRIDE_DECISION_20260918.md`); probe cells remain gate data and never enter the 640. The formal epoch confirms the probe: OFF collapses $0/40$ (Clopper–Pearson $[0, .0881]$) against the same frozen stratified baseline of .08602 over these 40 tasks, an observed gap of $-.0860$, negative in every (π, c_0) stratum.

Stop-loss and success. All six pre-specified P1 tests have their full pairings ($n = 20$ for the OFF contrasts, $n = 15$ for the oracle-divergent contrasts), with every paired difference nonzero and in the specified direction (exact one-sided Wilcoxon, raw $p = 2^{-20}$ and 2^{-15}; six-test Holm $p = 5.72 \times 10^{-6}$ and 1.22×10^{-4} respectively). Full reduces N_{call} and T relative to OFF by paired medians of 19.5 $[18, 20]$ calls and 18 $[17.5, 20]$ plan cycles; Fixed-7 minus Full is 3 $[1, 3]$ on both clocks, and Budget stops a paired median 1 $[1, 2]$ earlier than Full.

Table 15: Qwen3.5-9B FP16 P2 results on complete pairings. Differences and 95% paired bootstrap intervals are Full minus comparator. The OFF row is the separately specified two-sided harm screen.

Comparator	Pairs	Full succ.	Other succ.	p_{Holm}	Difference [95% CI]
Fixed-7	40	21	25	1.000	$-.1000$ $[-.2000, -.0250]$
Budget	40	21	17	.125	$+.1000$ $[.0250, .2000]$
OFF	40	21	40	—	$-.4750$ $[-.6250, -.3250]$

Both P2 superiority tests remain unestablished; discordant counts (Full-only/comparator-only success) are $0/4$ for Fixed-7 and $4/0$ for Budget. The OFF harm screen has $0/19$ discordances and two-sided exact $p = 3.81 \times 10^{-6}$; a budget-matched sensitivity for this screen appears in Appendix C.7. The arm ordering on fragile/faulted tasks, OFF $40/40 >$ Fixed-7 $25/40 >$ Full $21/40 >$ Budget $17/40$, coincides cell-for-cell with the primary epoch's counts, consistent with the seed-determined fault realizations and the deterministic trigger logic of Appendix C.6.

Collateral intervention. Clean success/valid counts are $80/80$, $80/80$, $78/80$, $80/80$ for OFF, Fixed-7, Budget, Full. Budget records two clean interventions (both final orders; .025, Clopper–Pearson $[.0030, .0874]$); the other governed arms record none (rule-of-three upper bound .0375). Reroutable successes are $20/20$, $19/20$, $19/20$, $20/20$, with 1, 2, and 3 recorded interventions in the Fixed-7, Budget, and Full arms. No trajectory ends at $T = 1$.

Mechanism of the lost pairs. All 19 Full-only losses end with `termination=completed`: the Governor enters wrap-up, issues a level-2 advisory, and the planner complies and closes the task without delivering. A spot check (task t005) resolves the trigger chain: the failure streak initiates the alarm ladder at the third consecutive failure, the online margin concurs one tick later ($\widehat{M}_k$ falls to $-.20$ by the wrap-up tick), and the dwell rule switches the regime at the sixth failure; the advisory

then ends the run. The same tasks succeed under OFF, which simply persists a few steps longer. On this deployment the advisory channel again operates as a stop command, and its cost falls on recoverable tasks.

Source and reproduction. Source [Q359B-ARM]: `phase0/results_arm640_qwen359b_20260918/`. The read-only entry point `phase0/analyze_qwen359b_completed.py` builds the 640-cell merged view over the eight shard directories and reuses the frozen pairing, parser, and statistical functions. The merged formal bundle digest is `bcdbacfe...c406`; the deployment epoch is `qwen35-9b-fp16-maslab8mini-20260918-a`. Each model epoch retains separate estimates and within-epoch intervals; all cross-model comparisons are descriptive.

C.14 Pre-registered transfer of the stopping law (fresh tasks, unified budget)

Role and design. The three FP16 epochs establish that stopping rules reallocate recovery opportunities; this epoch tests whether the *mechanism* behind that reallocation predicts outcomes on tasks the law has never seen. The pre-registration (`phase0/ARM640_TRANSFER_PREREGISTRATION_20260920.md`, frozen at commit `2ceac37`, SHA-256 `26dadceb...`; signed off and authorized for ignition 2026-09-20 04:37 +0800; ignition record `phase0/TRANSFER_IGNITION_RECORD_20260920.md`) fixes the design, the prediction table, and the decision rules before any run. Eighty fresh task seeds (`20260920000+i`) form three blocks: replication R (20 tasks, B=25, $\pi \in \{.8, .9\}$, including unsolvable and reroutable strata), persistence shift P (20 tasks, $\pi = .95$, B=25), and budget shift B (40 tasks, B=40). The four arms are verbatim the frozen ones; the only addition is a harness call cap, $\mathrm{HarnessCap}(B)$, applying the same executor-call budget to all four arms. The capped OFF arm is denoted OFF-B and must not be conflated with the uncapped OFF of the earlier epochs: the budget asymmetry discussed in Appendix C.7 is removed by construction here. The epoch forms its own statistical family and is never pooled with the earlier ones.

Deployment and validity. Serving used the incumbent 128k+tools vLLM 0.28.0 deployment of Qwen3.8-27B FP16 on the same host (GPUs 6/7), with the client-side clamp holding every request to prompt+completion ≤ 32768 tokens, identical to the frozen epochs' per-request semantics (deployment epoch `qwen38-27b-fp16-maslab8mini-20260920-t`). The frozen contract battery passes $11/14$ on this server; items 5a/5d/6, which assert a server-side 32768 limit, are structurally inapplicable to a 131072-context server and are recorded as known-not-applicable, with the clamp-evidence items 5b/5c passing. Deployment snapshots before ignition and after completion are identical except clocks and process tables (SHA-256 `c9bd8883...0689` / `7649c1ef...9ef5`). The matrix ran as eight concurrent driver shards split by budget (B=25: s0–s3; B=40: s4–s7) and completed $640/640$ cells contract-valid with zero invalid cells, zero breaks, and $8/8$ shard gate-audit passes in 7.4 hours.

The frozen law and its predictions. The mechanism law states that an arm loses exactly the recoverable tasks whose pre-recovery failure streak crosses the arm's effective threshold: $\Pr(\text{loss}) = \pi^{k_{\text{eff}}}$ with $k_{\text{eff}} = \max(k_{\text{switch}}, 3)$. Effective thresholds are Fixed-7: 7; Budget: $\max(9 - c_0, 0)$, i.e. $7/4/3$ at $c_0 = 2/5/10$; Full: $\lfloor(\kappa(B - c_0) - 2)/(1 + \kappa)\rfloor + 1$, i.e. $7/6/4$ at B=25 and $12/11/9$ at B=40; OFF-B: $B - c_0$. Before ignition, the prediction calculator was validated on the frozen 3.8 FP16 observations ($6/6$ arm$\times\pi$ cells inside their 95% intervals); the frozen 20-cell prediction table for the new design is `phase0/TRANSFER_PREDICTIONS_20260920.json` (SHA-256 `9326c610...`; the pre-registration's §10 records the content hash `9b64e289...` without the trailing newline, byte-identical by git diff).

E1: prediction adherence (primary). All $20/20$ (block $\times$ π $\times$ arm) cells have observed recoverable losses inside their frozen 95% prediction intervals (Table 16); the pre-registered criterion (≥ 18) is met, so the pre-registered verdict is that the law transfers. Global calibration $z = -0.78$. The three governed P-block cells sit together at their intervals' upper edge ($18/20$): the seeded stream gives $k_{\text{win}} \geq 7$ on 18 of the 20 seeds (expectation 14.0 under $\pi = .95$; interval upper bound 18), a seed effect shared across arms rather than an arm effect (post-hoc notes below). Per-cell discrimination is limited (post-hoc): $19/20$ intervals exclude at least one possible count, but only $13/20$ exclude at least half of the possible counts (mean coverage of the possible range .53); the $n = 5$ R-block cells

are widest (Fixed-7 at $\pi = .9$: $[0, 5]$), and each task's seeded fault stream is shared across arms, so cell outcomes are dependent. The task-level audit under "Compliance" below is the sharper test.

Table 16: Transfer epoch E1: observed recoverable-task losses against frozen predictions. Each cell gives observed losses, with the law's expectation and frozen 95% prediction interval in parentheses. All 20 cells land inside; the pre-registered criterion is ≥ 18.

Block (n per arm)	OFF-B	Fixed-7	Budget	Full
R, π=.8, B=25 (5)	0 (.10 [0,1])	0 (1.05 [0,3])	1 (2.05 [0,4])	0 (1.55 [0,4])
R, π=.9, B=25 (5)	1 (.74 [0,2])	1 (2.39 [0,5])	4 (3.25 [1,5])	2 (2.85 [1,5])
P, π=.95, B=25 (20)	9 (7.86 [4,12])	18 (13.97 [10,18])	18 (16.09 [12,19])	18 (15.23 [11,19])
B, π=.8, B=40 (15)	0 (.01 [0,0])	2 (3.15 [0,6])	5 (5.96 [3,10])	0 (1.51 [0,4])
B, π=.9, B=40 (15)	1 (.45 [0,2])	5 (7.17 [3,11])	7 (9.57 [6,13])	3 (5.02 [2,9])

E2: deterministic stop-loss boundary (primary). On governed arms $\times$ 15 unsolvable tasks (45 cells), the first wrap-up switch occurs within one executor call of the zero-LLM oracle's switch count on $45/45$ cells (44 exactly equal), meeting the pre-registered criterion ($\geq 41/45$); the target failure streak at the first wrap-up tick equals the oracle's switch streak on $45/45$. The oracle implements the pre-specified rule on the seeded streams, so this agreement validates trigger and budget accounting, not the rule's optimality. A post-hoc clarification: the analyzer's secondary (ii) item as implemented counted the streak at the first non-retry `advisory_L*` and reports $15/45$ only because Fixed-7 ($15/15$) and Full ($11/15$) are already at ladder level 3 at the switch, so the first wrap-up intervention is a `final_order` rather than an advisory; the pre-registered criterion above is unaffected. First wrap-up intervention kinds: Fixed-7 `final_order` 15; Full `final_order` 11 / `advisory_L1` 2 / `advisory_L2` 2; Budget `advisory_L1` 11 / `final_order` 4.

E3: budget adaptation of the Full arm. The law predicts Full's effective threshold moves $+5$ calls per c_0 stratum when the budget shifts to B=40 ($7/6/4 \rightarrow 12/11/9$). Observed unsolvable switch points move exactly $+5$ (N_{call} $9 \rightarrow 14$, $11 \rightarrow 16$, $14 \rightarrow 19$). As the law's corollary at the shifted budget, Full retains more recoverable successes than Fixed-7 in block B: $27/30$ vs $23/30$, paired difference $+.133$ $[+.033, +.267]$ (10,000 bootstrap, seed 20260920), with discordances Full-only 4 and Fixed-7-only 0 (the prediction: Fixed-7-only $= 0$ barring non-compliance). This is the same law that orders Fixed-7 above Full at B=25 in the primary epoch; it does not read as "DWELL outperforms". Post-hoc, on the same seeded streams and under the compliance audit below, fixed thresholds of $7/9/10/11/12$ consecutive failures would retain $23/26/26/27/27$ of the 30 block-B recoverable tasks: a threshold scaled with the budget ($7 \cdot 40/25 \approx 11$, or $7 + 5$) matches Full's $27/30$, so the block-B ordering reflects budget indexing rather than additional recovery information.

E4: unified-budget accounting. With all four arms under one call cap, Table 17 reports the faulted-workload accounting. Reading it as one utility problem, $U_a(v) = vS_a - C_a$ with v the token value of one incremental success, the utility-optimal arm is Budget for v below $\approx$49,000 tokens, Full for $v \in [49{,}000, 193{,}500]$, and OFF-B above; Fixed-7 has no optimal interval on this workload. In the P block all governed arms deliver only $2/20$ successes at 259,128–353,784 tokens per success: at $\pi = .95$ governance is strictly harmful to recoverable tasks, as the law predicts—a predicted operating point, not a new counterexample.

E5: safety. Clean tasks succeed $80/80$ in every arm with zero interventions and zero cap hits; reroutable/faulted tasks succeed $5/5$ in every arm with zero interventions.

Compliance and the single concession. All 84 lost governed fragile runs satisfy the seeded $k_{\text{win}} \geq k_{\text{eff}}$ (zero early concessions), and no run makes a target call after the first wrap-up intervention. Conversely, no retained governed run reaches its threshold (post-hoc): task-level outcomes match the rule "lost iff $k_{\text{win}} \geq k_{\text{eff}}$" on $180/180$ governed fragile runs (150 with a seeded recovery inside the cap and 30 on the ten tasks whose seeded stream never recovers) and on $59/60$ OFF-B runs, the exception being t065 below. OFF-B loses 11 fragile runs: ten are harness cap hits; one (task t065, block B, $\pi = .9$, B=40) is a genuine early concession—seeded $k_{\text{win}} = 17$, the planner stopped after 12 consecutive target failures at $N_{\text{call}} = 22$, below the 40-call cap. This is the epoch's only

Table 17: Transfer epoch E4: faulted-workload accounting under the unified harness call cap. OFF-B cap hits are truncations at the shared cap; governed arms never hit it. Totals feed the utility analysis in the text.

Block	Arm	n	Succ.	Tokens	Tokens/succ.	Cap hits
R	OFF-B	20	14	918,545	65,610	6
R	Fixed-7	20	14	571,442	40,817	0
R	Budget	20	10	486,750	48,675	0
R	Full	20	13	523,385	40,260	0
P	OFF-B	20	11	1,415,698	128,700	9
P	Fixed-7	20	2	707,567	353,784	0
P	Budget	20	2	518,257	259,128	0
P	Full	20	2	594,118	297,059	0
B	OFF-B	40	29	2,435,769	83,992	7
B	Fixed-7	40	23	1,104,782	48,034	0
B	Budget	40	18	859,880	47,771	0
B	Full	40	27	1,332,950	49,369	0
All	OFF-B	80	54	4,770,012	—	22
All	Fixed-7	80	39	2,383,791	—	0
All	Budget	80	30	1,864,887	—	0
All	Full	80	42	2,450,453	—	0

behavioral fragility-gap event ($1/60$ OFF-B fragile runs) and is reported as a single case without statistical claims.

Post-hoc clarifications. Written after the pre-registered report; no endpoint changed. (i) The pre-registration's §10 hash records the prediction file's content hash without the trailing newline; the analyzed file is byte-identical to the frozen one. (ii) The E2 secondary item's implementation artifact, above. (iii) The P-block shared upper-edge placement, above. (iv) t065, above.

Source and reproduction. Source [TRANSFER-ARM]: results (with `SHA256SUMS`) in the transfer worktree, branch `transfer-20260920`, `phase0/results_transfer_20260920/` (640-file bundle SHA-256 `33cf9cbf...947b`); raw shards `../agent-transfer/phase0/stress_data_transfer_qwen38fp16_20260920_s\{0..7\}/`. The read-only entry point `phase0/analyze_transfer_20260920.py` re-derives every table here from the raw trajectories and refuses to run until all 640 cells are valid; the analysis script and the prediction file are byte-identical to their pre-ignition frozen versions.

D Extended Related Work

Agent evaluation and failure analysis. ReAct and Reflexion established widely used patterns for interleaving language reasoning, actions, and verbal feedback (Yao et al., 2023; Shinn et al., 2023). AgentBench and AgentBoard evaluate interactive agents and expose progress signals (Liu et al., 2024; Ma et al., 2024); MAST catalogs specification, coordination, verification, and termination failures in multi-agent systems (Cemri et al., 2025). The ARE environment and Gaia2 benchmark include noisy scenarios with tool failures and irrelevant events (Froger et al., 2025). ReliabilityBench frames reliability as a surface over repetition (pass^k), perturbation intensity ϵ, and injected fault intensity λ (Gupta, 2026). Our persistence and redundancy sweeps connect controlled fault conditions to an online measure of task-potential contraction and the consequences of budget-aware intervention.

Process signals and runtime recovery. Process supervision evaluates intermediate reasoning (Lightman et al., 2024). Entropy dynamics characterize centralized orchestrators and introduce the IWG benchmark used in our external evaluation (Zhu et al., 2026). Stable agentic control gives formal guarantees by constraining an LLM to deterministic tools and finite action catalogs in cyber defense (Prinos et al., 2026). Self-healing orchestrators diagnose failures and apply budgeted recovery actions (Babu & Agrawal, 2026). BENCH2ROBUST studies recovery under controlled tool errors using structured Bayesian Tool Memory and reinforcement learning (Chen et al., 2026). Its recovery context summarizes historical rollouts and fallback options. We study a trajectory-level progress

signal coupled to an explicit retry-to-wrap-up rule based on current failure history and remaining budget.

Budget awareness and agent early stopping. BAGEN elicits remaining-budget intervals and infeasibility predictions from trajectory prefixes, evaluating the associated stopping decisions with a rollout–replay protocol (Lin et al., 2026). Ruan et al. (2026) develop hidden-state abort cascades with calibrated control of successful-episode recall. These works establish direct precedents for using predicted feasibility and outcomes to limit agent expenditure. DWELL uses in-band executor traffic and a specified recovery model to govern live retries under tool faults. Our four-arm evaluation measures both savings on unsolvable tasks and successes lost on recoverable ones; our stall-alarm guarantee does not provide a success-recall guarantee.

Metareasoning, stopping, and restarts. Rational metareasoning evaluates computation through its expected effect on action quality and the cost of delaying action (Russell & Wefald, 1991). Bayesian metalevel selection models formalize whether to acquire more evidence or stop and select an action (Hay et al., 2012). Restart theory allocates runtime across independent executions of Las Vegas algorithms (Luby et al., 1993). These are foundations for valuing additional effort. DWELL governs executor retries within an ongoing trajectory, retaining accumulated task state. Its posterior predictive threshold defines a concrete governance policy; it is not a solution to a general optimal-stopping problem.

Switched-system stability. Even stable modes can produce an unstable switched system when switching is too fast; average dwell-time theory bounds this effect (Hespanha & Morse, 1999; Liberzon, 2003). This perspective motivates $\mathcal{M}$ as a summary of observed contraction and switching effects, and DWELL as a state-dependent retry dwell with a budget-aware stopping rule. Our formal claims concern the stated trace-level alarm and controller-accounting assumptions.