COST-INDEXED BRACKETING FOR CHECKABLE SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

We study hard-budget search with a sound terminal checker and a supplied finite hidden-state model. Stationary rejection-only retries reduce to completion counts; an exit-preserving phase construction extends this reduction through checkpoint creation. The resulting block success vectors are standard partially observable Markov decision process (POMDP) α-vectors, with exact cost expressed by observable budget augmentation. We show how a cost-sliced envelope approximation composes through diagnostic and refresh decisions, and implement both rounded grids and standard ε-pruning. The latter is substantially more compact on the strongest tested bank: 255 vectors rather than 31,083 grid records. Computed lower/upper brackets, not a loose worst-case grid rate, guide refinement. A finite-row calibration procedure produces a true-environment bracket from sampled program proposals under explicit labeled-access assumptions. Experiments include measured 10,000-prior workloads, support-size resource failures, and 64 random graph catalogs. Post-refresh diagnosis helps modestly in unchecked catalogs but remains of little benefit when refresh already checks the checkpoint; greedy residual filling explains most fixed-bank allocation gains. The contribution is an exit-preserving search specialization of established POMDP approximation tools, with an explicit model-acquisition boundary, not a new value representation or a general claim about language-model reasoning.

1 INTRODUCTION

A solver that can check its answer exactly still has to decide how to search. After rejection, it may retry a saved fragment, inspect it, or generate a new checkpoint. Several attempts from the same mistaken fragment are not independent opportunities to succeed. Yet a rejection bit is also not a counterexample: only the latter may change which fragments or tests become available. We study this distinction for **hard-budget search with a sound terminal checker and a supplied finite hidden-state model**. The applications are bounded specification repair and controlled constraint search, not unrestricted language-model reasoning.

Our starting question is structural: which parts of an observation-dependent search policy can be replaced by a precomputed block? With stationary rejection-only feedback, a deterministic policy has one all-failure path; its completion counts suffice. Stationary diagnostics can be moved before that block. When generation creates a checkpoint or returns informative failure content, this transformation must stop. Preserving the *full unsuccessful exit law*, including remaining resources and the next hidden state, lets the resulting blocks compose across such refreshes.

The right comparison for compressing these blocks is a partially observable Markov decision process (POMDP), not merely a stochastic knapsack. A block's statewise success probabilities are α-vectors. Their reuse across beliefs, linearity, and approximate dominance are established ideas (Smallwood & Sondik, 1973; Cassandra et al., 1997; Cassandra, n.d.). Adding observable remaining budget to the state also expresses exact-cost indexing. We therefore do *not* claim a new value representation or pruning principle. We develop an exit-preserving specialization: precompute stationary completion blocks at each exact cost, then use ordinary α-vector pruning or a rounded failure grid inside the same diagnostic/refresh recursion. An error charged to each block composes under a single hard-budget allowance.

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.

This positioning changes the algorithm. On a moderate two-state bank with 32 completion actions and budget 24, an ε-pruned library keeps 255 vectors, versus 31,083 grid records, and gives a substantially narrower bracket. We use the stronger representation rather than treating it only as an inconvenient baseline. Continuous-count relaxation remains a useful competitor for a terminal query. None of these comparisons establishes superiority to optimized general-purpose POMDP software; our implementations isolate the representation within the same search interface.

A narrow fitted-model bracket is not a certificate for an unknown environment. We therefore test the full process in a finite specification-repair example: sample every declared completion row, obtain simultaneous binomial confidence intervals, and propagate them through planning. The resulting environment-level interval has a stated coverage guarantee. This requires queryable labeled modes and known diagnostic/refresh laws; it does not learn sufficient hidden states or certify an unobserved abstraction.

The experiments separate three questions. First, how do representations trade preparation, query cost, and bracket width? Second, does a learned refresh controller help on random catalogs when compared with the same-feedback greedy and globally frontloaded alternatives? Third, how wide is an environment-level bracket once sampling error is included? We report measured 10,000-query sequences, support-size failures, 64 unfiltered graph catalogs, and actual program-proposal samples. Results are selective: post-refresh diagnosis helps modestly in unchecked catalogs, but is of little benefit when refresh already checks the checkpoint. Simple residual filling explains most fixed-bank gains. The contribution is a focused planning and validation method, not a claim that complicated search is generally necessary.

2 Search interface and an exit-preserving normal form

State, actions, and costs. The observed state is a bank x of saved prefixes, remaining budget b, and diagnostic/refresh quotas k, r. The hidden state $\ell \in \{1, \dots, K\}$ is sufficient for the following kernels. All action costs and the initial budget B are **positive integers**, except that b may reach zero. A plain completion d costs $c_d(x)$, succeeds with probability $s_d(\ell, x)$, and otherwise reveals only rejection, leaving the bank and hidden state unchanged. A sound terminal check is included in its cost; success is absorbing. A paid diagnostic j has kernel $K_j(y \mid \ell, x)$ and cost $a_j(x)$. Within a phase, these outcomes are conditionally independent across invocations and their laws are order-independent. A deterministic diagnostic is included by putting the task in ℓ.

A refresh h costs $a_h(x)$, succeeds with probability $r_h(\ell, x)$, or gives observed outcome z, bank $x_{h,z}$, and next hidden state according to a nonsuccess kernel $T_{h,z}(\ell, \ell'; x)$. Its row sums plus r_h equal one. Retaining a new fragment or revealing a counterexample is a refresh, not free information on a plain failure. Kernels cannot depend on omitted history. Persistent diagnostic errors require persistent state coordinates, not repeated independent noise draws. Our objective is the probability of *discovering a checked solution before spending* B; it is not answer selection with an imperfect verifier. The initial bank is supplied and its already-spent construction cost is common to all policies.

Let $u \geq 0$ be an unnormalized hidden-state mass, $\mathcal{N}_t(x) = \{n \in \mathbb{N}_0^{m_x} : c(x)^\top n = t\}$, and

$$f_n(\ell, x) = \prod_d [1 - s_d(\ell, x)]^{n_d}, \qquad \alpha_n = \mathbf{1} - f_n. \tag{1}$$

Here f_n is the probability that all planned completions fail; $0^0 = 1$. A count plan can stop at its first success in any chosen order.

Theorem 1 (Failure-only allocation)**.** *Without diagnostics or refreshes, every deterministic policy has the same statewise deadline success as a count plan. The optimum is* $\max_{t \leq B, n \in \mathcal{N}_t} u^\top \alpha_n$. *Randomization cannot improve it.*

Removing success leaves produces one all-failure path; its counts give (1). Failures can update beliefs, but they do not create alternative rejection-only histories. This elementary observation is not a general claim that feedback is unnecessary.

Theorem 2 (Diagnostic frontloading)**.** *Within a stationary bank, every feasible policy has an equivalent form: adaptive diagnostics, then a count block. Statewise deadline success, worst-case budget, and diagnostic quota are preserved. Expected expenditure need not be preserved.*

Delete success leaves, contract unary completion nodes, and record their counts at diagnostic leaves. Each all-failure path has the same diagnostic-likelihood and failure factors after reordering. Tests are already available, and total cost on that path is unchanged. Appendix A gives the complete argument.

Theorem 3 (Exit-preserving phase composition)**.** *Under the finite stationary-phase interface, every feasible policy has the following equivalent form in each phase: adaptive diagnostics, a count block, and a selected refresh or stop. The transformation preserves phase success and the joint law of every unsuccessful exit: next hidden state, observations, bank, remaining budget, and quotas. Recursively composing the transformed phases preserves total deadline success. Diagnostics are not moved across a refresh.*

For diagnostic path γ, its likelihood $A_\gamma(\ell)$, block n, and refresh h, the mass of exit (z, ℓ') is $A_\gamma(\ell) f_n(\ell) T_{h,z}(\ell, \ell')$ in either ordering. Exact remaining resources also agree. Induction over refreshes proves the claim; equal immediate success alone would not suffice. These are *sufficient conditions*, not a necessary-and-sufficient characterization of dispensable adaptation. If every failed generation changes the proposal or bank, every generation may be a refresh and little is reduced.

The phase recursion maximizes over: a diagnostic now; a block and stop; or a block and refresh. With $J_{k,r}(b, x, u)$ denoting optimal success mass, the latter choice has value

$$u^\top (1 - f_n) + (u \odot f_n)^\top r_h + \sum_z J_{k,r-1}\big(b - t - a_h, x_{h,z}, T_{h,z}^\top (u \odot f_n)\big)\,. \tag{2}$$

An affordable diagnostic contributes $\sum_y J_{k-1,r}(b - a_j, x, u \odot K_j(y \mid \cdot, x))$. Positive costs make recursion finite. Exact counts may nevertheless be numerous, and the number of reachable beliefs is not polynomially bounded. Appendix A specifies all boundary cases.

3 The relationship to POMDP α-vectors

What is represented. In a POMDP, a fixed policy has a value vector indexed by the hidden state; the best value is the upper envelope of those vectors over beliefs (Smallwood & Sondik, 1973). Equation (1) is exactly such a vector for a completion block. For a fixed permitted continuation with statewise success $h \in [0, 1]^K$ after that block,

$$G_h(u, f) = u^\top h + [u \odot (1 - h)]^\top \alpha, \qquad \alpha = 1 - f. \tag{3}$$

The reweighted vector is nonnegative and has mass at most $\|u\|_1$. Consequently, ordinary envelope ε-dominance already controls *every* such continuation. One need not find a single representative within sup-norm ε of each vector. An envelope can approximate a vector using different representatives at different beliefs; this weaker condition is often much cheaper.

Budget augmentation does express exact cost. Add observable remaining budget to the POMDP state, and disallow unaffordable actions. A completed block of exact expenditure t exits at the observable coordinate $b - t$, so budget-sensitive continuations are expressed without any new formalism. Because budget is observed, beliefs occupy K-dimensional slices, not an arbitrary simplex of dimension $K(B + 1)$. A uniform coordinate grid has at most $(Q + 1)^K$ cells per slice and $(B + 1)(Q + 1)^K$ across costs. This is also our grid bound; it is *not* an improvement over an appropriately budget-augmented α construction.

What is inherited, and what is specialized. Piecewise-linear value functions, linear-programming dominance, ε-pruning, and belief-point approximation are established (Cassandra et al., 1997; Cassandra, n.d.; Pineau et al., 2003; 2006; Lovejoy, 1991). Succinct multiobjective approximations also predate this work (Papadimitriou & Yannakakis, 2000). Our specialization is the exit-preserving block normal form, an interchangeable *exact-cost block oracle*, and the accounting that composes its errors through search refreshes. We prove below that standard α pruning can replace the grid in that oracle. This is an application and composition of existing representation ideas, not a claim to have discovered belief-independent vectors or exact-cost state augmentation. Point-based methods may be preferable on large reachable-belief spaces; we do not reproduce a complete external POMDP package or claim dominance over one.

Table 1: Representations compared at a fixed observed bank. Exact-cost slicing is available in the standard budget-augmented formulation. α pruning and point-based approximation are prior techniques, not contributions of this paper.

Representation	What is kept	Guarantee / coverage	Size dependence
Cost-sliced α pruning	Executable success vectors at each exact cost	Envelope loss over the full belief simplex	Instance-dependent; potentially exponential
Failure grid	Executable vector and lower failure envelope per (t, k)	Statewise grid error; hence envelope error	At most $(B+1)(Q+1)^K$
Point-based POMDP methods	Vectors chosen for representative beliefs	Depends on belief coverage and backup error	Selected belief set, not universal grid bound

4 Two block oracles and the bracket used in practice

Rounded grid: a finite reference construction. At exact spent cost t, store a rounded failure vector $k \in \{0, \dots, Q\}^K$, an executable count vector, and its actual failure vector. Starting at $(0, Q\mathbf{1})$, appending completion d sends k_ℓ to $\lceil (1 - s_d(\ell))k_\ell \rceil$ and adds c_d to t. Merge identical (t, k) keys. A path with j attempts satisfies $0 \le k/Q - f \le j\mathbf{1}/Q$. All paths at a key therefore lie in the same known failure box. Coordinatewise minima over those paths give a valid optimistic vector, but need not be executable together.

Theorem 4 (Grid finiteness and phase termination)**.** *Assume positive integer completion, diagnostic, and refresh costs, integer budget B, and known sufficient kernels. Let c_* be the minimum completion cost* across all reachable banks*. Each bank's grid has at most $(B+1)(Q+1)^K$ records and at most $m(B+1)(Q+1)^K$ attempted extensions. Actual representatives give an executable lower value L_Q; optimistic failure envelopes give $U_Q \ge V^*$. For normalized priors,*

$$0 \le U_Q - L_Q \le \min\{1, \lfloor B/c_* \rfloor / Q\}. \tag{4}$$

This is a worst-case finiteness and refinement guarantee, not the practical precision estimate. Our stopping test uses the *computed* interval, usually much narrower. For example, on the moderate $(m, K, B) = (32, 2, 24)$ bank, $Q = 128$ gives a 9.38-point worst-case bound but a 0.95-point maximum computed gap on 32 priors. The implementation uses propagated coordinate minima; a bank-specific $c_{*,x}$ may tighten an analytic local envelope, but (4) always uses the global minimum. Grid order and representative selection need not be nested in Q. The running lower maximum and upper minimum remain valid even when an individual refinement worsens.

Standard α pruning as a stronger oracle. Let $\mathcal{A}_t$ be all success vectors at exact cost t. Keep an executable subset $\widehat{\mathcal{A}}_t$ satisfying

$$\max_{\alpha \in \mathcal{A}_t} v^\top \alpha \le \max_{\beta \in \widehat{\mathcal{A}}_t} v^\top \beta + e_t \|v\|_1 \quad (v \ge 0). \tag{5}$$

Appending d backs up α to $s_d + (1 - s_d) \odot \alpha$. We omit a candidate only if a linear program (LP) or a componentwise witness proves its envelope advantage at most ρ. Using exact cost, local loss certificates accumulate into $e_t \le \lfloor t/c_* \rfloor \rho$. Retained vectors are executable; their number is instance-dependent. A feasible dual mixture, rather than a primal LP objective alone, upper-bounds each omitted advantage in the numerical implementation.

Theorem 5 (An envelope oracle composes through refreshes)**.** *Suppose executable cost-sliced sets satisfy* (5)*, with $e_t \le \lfloor t/c_* \rfloor \rho$. Evaluate* (2) *using these sets for the lower planner. In the upper planner add $e_t \|u\|_1$ to a block's value, propagate upper continuation values, and cap by $\|u\|_1$. Then*

$$L_\alpha \le V^* \le U_\alpha, \qquad U_\alpha - L_\alpha \le \min\{1, \lfloor B/c_* \rfloor \rho\}. \tag{6}$$

No additional multiplier for the number of diagnostic branches or refreshes is needed.

Equation (3) applies (5) to an exact optimal continuation, even if that continuation was selected after observing a failure. Exact cost preserves its affordability. The upper recursion pays for the local envelope loss; its subsequent loss is bounded by remaining failure mass and remaining completion work. Diagnostic outcome masses sum to the parent mass. Induction gives (6). Appendix B proves both pruning and composition, including why clipping each optimistic vector's coordinates separately would not justify this scalar-error algorithm.

How a bracket is used. Preparation and stopping are separate decisions. For a known terminal model, first evaluate an inexpensive executable plan and a cheap upper bound. If their gap meets the requested tolerance, no library is needed. Otherwise choose a representation according to anticipated reuse: the grid has an explicit worst-case record bound, whereas LP pruning exploits the actual envelope and can be much smaller. A phase query needs an upper bound valid for that entire diagnostic/refresh interface; a terminal-only bound is not silently substituted for it.

Each completed computation returns an executable policy with lower value L_j and a valid upper value U_j. The running interval is

$$[\max_{j\le t} L_j,\ \min_{j\le t} U_j\,].$$

Retain the policy that supplied the largest lower endpoint. Its success is guaranteed by that endpoint, even if the smallest upper endpoint came from a different representation. This makes interruption and nonnested refinement useful: partial computations are not discarded merely because a later grid is worse. The outer procedure stops when the computed gap is acceptable or reports a resource-limited interval. The a priori rates prove eventual precision for finite known models, but are not used as substitutes for measured preparation cost. With estimated models, the same intersection is valid only after every included endpoint has been justified on a common confidence event, as in Section 5.

Cheap terminal bounds remain important. Observing the hidden state before choosing an allocation gives an upper bound obtainable by statewise integer knapsacks. For strictly interior rates, continuous counts minimize the convex failure objective $\sum_\ell w_\ell \exp(\sum_d n_d \log(1 - s_{d\ell}))$ over $n \ge 0, c^\top n \le B$ (Boyd & Vandenberghe, 2004). A global tangent lower bound on failure, not the optimizer's feasible objective, supplies an upper bound on success. Compare it with a separately charged greedy incumbent. Scalar terminal bounds do not themselves provide a reusable open-exit block oracle; this does not rule out stronger relaxation-based phase methods. Appendix F gives formulas and numerical-witness checks.

5 When a model bracket is an environment certificate

Declared rows can be calibrated. Assume a *specified finite* hidden-state family and a sampling interface for each labeled completion row (x, d, ℓ). Diagnostic and refresh laws are known. From R_i conditionally independent trials in row i, form simultaneous Clopper–Pearson intervals $[l_i, u_i]$ with total failure probability δ (Clopper & Pearson, 1934). Duplicate rows shared by banks need only one interval. A fixed finite collection of sample sizes is included in the same union bound, allowing data-dependent choice among those sizes without spending coverage again.

Theorem 6 (Acquired finite-row brackets)**.** *On the simultaneous interval event, replace completion rates by their lower and upper endpoints. An executable lower-model planner with value L^- and an upper-model planner with value U^+ satisfy*

$$L^- \le V_{\mathrm{true}}(\pi_-) \le V^*_{\mathrm{true}} \le U^+. \tag{7}$$

Thus the interval has coverage at least $1-\delta$, including for planners and recorded resolutions selected from the calibration data. Alternatively, let $\widehat{s}_i$ be a fitted rate and $\eta_d = \max_\ell \max\{\widehat{s}_i - l_i, u_i - \widehat{s}_i\}$. These are acquired uniform bounds on the corresponding Bernoulli kernel errors; a budget-weighted coupling transfers a nominal bracket as $[L - E_B, U + E_B] \cap [0, 1]$.

For a fixed public-history policy, increasing a completion success probability cannot hurt: the immediate-success value is one and the failure continuation is at most one. Induction proves monotonicity, yielding (7). For the alternative bound, set $D_0 = 0$ and $D_b = \max_{a:c_a\le b}\{\eta_a + (1 - \eta_a)D_{b-c_a}\}$, with $\eta_a = 0$ for known kernels; $E_B = D_B$ for an exact initial prior. Appendix D proves the simultaneous statement. The rate-box bound is usually tighter than generic coupling here.

An end-to-end reporting rule. The acquisition and planning steps can now be stated as a procedure. Declare which modes and action rows calibration can query; record which other kernels are supplied. Construct one simultaneous confidence event covering every row and sample size that may be used. Plan a feasible policy under lower completion endpoints and bound all policies under upper endpoints. Report both the data confidence level and the numerical planning gap, together with the acquisition cost. Refining a library reduces only planning approximation; it does not narrow the row intervals.

Table 2: Guarantees and experiments concern different information regimes. Numerical agreement tests implementations; it is not a replacement for a proof.

Result	Required model information	Supporting study
Theorems 1–3	Stationary phases; sufficient kernels	Independent primitive/phase recursion
Theorems 4–5	Known finite support and kernels	Grid/α/convex checks, support limits
Theorem 6	Declared queryable rows; known other laws	Sampled decoder with simultaneous intervals
Learned controller comparison	Fitted rates; three supplied layouts	64 random catalogs; *not* a true-model certificate

Buying more samples may reduce the latter, but an unlisted adaptive sample size requires additional statistical accounting.

This distinction matters for the graph experiment. Its smoothed 128-rollout point estimates and across-catalog standard errors describe policy performance, not a uniform error bound for every action kernel. We do not convert those standard errors into certification radii. In the decoder, every uncertain row is explicitly queried and all five sampling levels are covered at once, so a policy selected after looking at those data is included in the same event. The smaller number of unique proposal menus also matters: one semantically identical menu can share a row estimate across several observed banks. Such sharing follows from a declared equality of kernels, not from assuming that visually similar checkpoints have the same success probability.

What this cannot learn. Finite-row intervals do not discover omitted persistent variables, validate a partition of an unknown fine state space, or provide uniform coverage on unqueried rows. Approximate aggregation additionally requires a uniform bound on the *joint public-output and next-aggregate kernel* at every fine state, not merely small mean prediction error (Ferns et al., 2004). Without such a supplied or independently estimable condition, a bracket certifies only the fitted model. Our fine-to-aggregate coupling theorem and a counterexample to average-rate substitution are in Appendix E.

A complete, small specification family. The repair task decodes two bytes. Target modes differ only in byte order and signedness, giving four states by construction. Sixteen candidate implementations also vary an offset. Probes return mode-dependent outputs; refreshes retain another code fragment. The exact checker evaluates each candidate on all 65,536 byte pairs. Conditional on the mode and bank, specified proposal and probe laws have no residual hidden-history dependence. Deployment does not observe the mode. Calibration can query each declared mode against its reference checker, but receives sampled proposal outcomes rather than the proposal distribution's success rates. Unknown widths, undocumented encodings, or independent persistent proposal faults would invalidate the four-state model. This is a closed-family proof of an estimable certificate, not a broad program-repair benchmark.

6 Experiments: representation, adaptation, and acquisition

Costs are controlled generation, checking, and probing work units, not measured large language model (LLM) tokens. Wall-clock planning and calibration are separately reported. Exact integration at test time removes rollout noise but gives no oracle rate to the controller. Table 2 maps claims to assumptions and evidence; full configurations and failed limits appear in Appendix H.

Representations, not only executable baselines. On the completed terminal grid, convex relaxation is tighter than the failure library in 38 of 70 cells, the library in five, with 27 ties; shared banks mean these are not 70 independent tasks. We compare cost-sliced α pruning with per-backup allowance $\rho = .01/\lfloor B/c_* \rfloor$. Eleven of twelve bank comparisons finish within their 25-second cap; the largest three-state comparison times out during the grid portion. On the moderate bank $(m, K, B) = (32, 2, 24)$, the grid at $Q = 128$ stores 31,083 records and has maximum gap 0.95

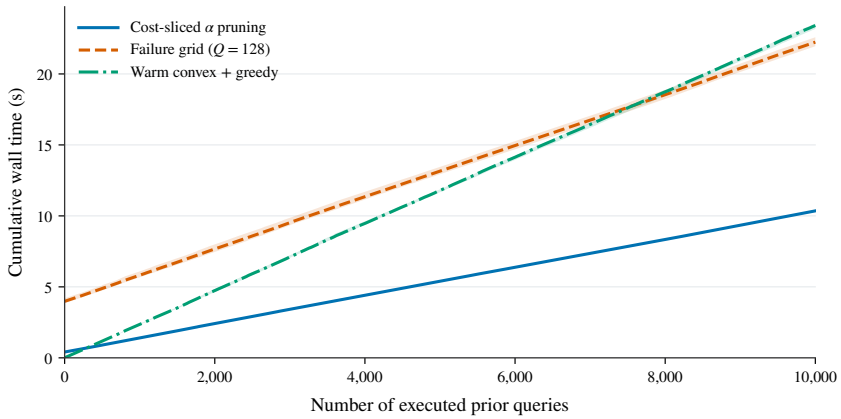


Figure 1: Measured cumulative cost on 10,000 priors; three serial timing repeats of the same workload. Lines show the repeat mean and shaded bands the repeat minimum and maximum, not confidence intervals. Every method includes the separately computed greedy incumbent. All meet a one-percentage-point tolerance on every measured query. This is one known moderate bank, not a universal amortization law.

percentage points over 32 priors. α pruning stores 255 records and gives 0.10 points; the convex-plus-greedy gap is 0.21 points. Approximate α sets provide a stronger comparison on this bank.

The same oracle works in the two-bank phase workload. With 32 actions per bank, $B = 24$, one diagnostic and one stochastic refresh, there are 16,855,983 feasible counts per bank. The α-pruned phase planner uses 685 records across both banks and brackets success by $[95.06\%, 95.38\%]$. Building both lower and upper planners takes 2.04 seconds; solving their recursions takes 37.00 milliseconds in the recorded run. This includes 77,498 scored block choices, not just memoized states. All 80 smaller random brackets contain an independent primitive optimum; executing each lower policy reproduces its claimed value. The result is scalable planning in a small known model, not a runtime claim against an unexecuted exhaustive solve.

Measured reuse and support-size limits. We executed 10,000 prior queries on the moderate bank, three times. Including a separately computed greedy incumbent for every method, total times range from 10.28–10.41 seconds for α pruning, 21.95–22.59 for the grid, and 23.17–23.55 for warm convex solves. All three meet one percentage point on every measured query. Final grid/convex crossings occur at 7,215–8,027 *executed* queries; no extrapolated curves are used. These repeats measure runtime variability, not independent performance replications.

Support scaling uses the same 16-action family, $K = 2, 3, 4, 6, 8, 12$, budgets 12 and 24, and a one-point target over 32 declared priors. At $B = 24$, α pruning reaches $K = 6$ with 1,843 vectors, 7.14 seconds, and a 0.57-point maximum gap. $K = 8$ and 12 exceed 20 seconds. The grid finishes at $K = 3$ with 264,181 records and a 0.89-point gap but fails its resource/tolerance target at larger supports. At $B = 12$, $K = 12$ is feasible, but there are only 650 possible counts: that is a small-budget case, not large-search scalability. Table 6 records all successes and failures.

Random checkpoint catalogs. The study uses independent width-three layered graphs at four fixed edge densities, three labeled layouts per catalog, four initial completion actions, banks growing to eight, and up to two diagnostics/refreshes. We use 64 consecutive catalog seeds, 16 per density, without selecting on performance. The eight catalogs used for action-level checks and the remaining 56 are also analyzed separately. Collision-free, namespaced Philox streams preserve intentional within-catalog sharing across methods and sample sizes. The study uses 354,048 distinct maximal-budget completions and 1,920 policy/configuration evaluations. Rates use 128 samples per declared row; diagnostic and refresh laws are supplied. This is within-catalog model fitting, not unseen-layout transfer or an environment certificate.

The unchecked interface exposes checkpoint creation without revealing validity for free. Here post-refresh diagnosis gives mean Phase-minus-Global gains of 0.43–0.56 points (Table 3). In the 56

Table 3: All six catalog conditions. Phase is a success percentage; differences are percentage points relative to same-feedback baselines. Errors are *stratified standard errors*, $\sqrt{\sum_h s_h^2/n_h}/4$, over four fixed density strata with 16 independent catalogs each; they are not confidence intervals. Global disallows diagnosis after the first refresh. Greedy and One-step use different tail approximations, checked below.

Interface / (B, k, r)	Phase	Δ Frozen	Δ Global	Δ Greedy	Δ One-step
Checked (18,1,1)	53.17	7.68 ± 0.76	0.02 ± 0.02	0.64 ± 0.16	0.59 ± 0.14
Checked (18,2,2)	55.43	9.79 ± 0.88	0.04 ± 0.03	0.95 ± 0.24	0.92 ± 0.23
Checked (30,2,1)	64.54	7.30 ± 0.76	0.08 ± 0.05	0.83 ± 0.22	0.82 ± 0.22
Unchecked (18,1,1)	54.18	8.70 ± 0.79	0.43 ± 0.13	1.35 ± 0.34	1.32 ± 0.34
Unchecked (18,2,2)	56.54	10.90 ± 0.92	0.56 ± 0.13	1.60 ± 0.32	1.55 ± 0.32
Unchecked (30,2,1)	65.88	8.64 ± 0.83	0.51 ± 0.14	0.89 ± 0.23	0.79 ± 0.23

Table 4: Data-acquired environment intervals, all in percentages. Both bracket types have simultaneous at-least-95% coverage under the declared finite-row sampling model. "TV" uses the *acquired* row radii with known other kernels, not declared perturbation levels. True optimum is 61.25%; the last column independently evaluates the executable rate-box lower policy. Nested sample sizes are not independent experiments.

R per row	Total draws	Acquired-TV bracket	Rate-box bracket	True policy
256	7,168	[9.69, 100.00]	[50.48, 71.03]	61.04
1,024	28,672	[32.16, 90.81]	[56.30, 66.59]	61.04
4,096	114,688	[45.97, 77.31]	[59.07, 64.17]	61.25
16,384	458,752	[53.09, 69.28]	[59.91, 62.45]	61.25
65,536	1,835,008	[57.22, 65.46]	[60.71, 61.97]	61.25

catalogs outside the action-check subset, the $(18, 2, 2)$ difference is 0.59 points, with a descriptive stratified bootstrap interval $[0.33, 0.87]$. Checked refreshes already reveal validity and remain nearly null. We report every condition and avoid interpreting the deliberate 1 versus $1/2$ separation as typical behavior. Improvements against receding greedy are modest, not a general superiority claim.

Identical scores, action checks, and null results. On the eight-catalog action-check subset, Frozen purchases exactly one diagnostic in seven at budget 18 and six at budget 30; it never buys a second. That explains equal Frozen values when only diagnostic quota increases. Greedy and One-step have identical reachable policies in four to seven of eight catalogs, depending on the condition. Coupled first-disagreement probabilities range from 1.13% to 19.95%, so equal rounded values are not proof of universal policy identity. On the separate 18-condition fixed-bank Boolean satisfiability (SAT)/graph study, residual filling ties diagnose-then-allocate (DTA) in 14 conditions, including every graph condition. SAT absolute successes are often only 2–17%; sub-point gains there do not support broad practical importance. The expected-cost compiler gives no diagnostic deferral/omission saving on 144 fitted policies. This control distinguishes expected expenditure from deadline success.

An acquired true-environment bracket. For the decoder, seven proposal menus and four modes give 28 queryable rows. We sample actual candidate programs and check them; true rates are used only by a separate evaluator. Five nested sample sizes are covered jointly by 140 Clopper–Pearson intervals with $\delta = .05$. At 65,536 draws per row, 1,835,008 distinct maximal-budget proposals give a true-environment rate-box bracket $[60.71\%, 61.97\%]$. The deployed lower policy's true success and the exact optimum are both 61.25%. With 4,096 draws per row, the bracket is wider but non-vacuous: $[59.07\%, 64.17\%]$. The more general acquired total variation (TV) transport also works but is substantially looser (Table 4). The example closes a finite-row acquisition loop; it does not solve unknown-support abstraction or demonstrate that sampling is cheaper than exact analysis of this tiny grammar.

Diagnosing non-monotone refinement. The reported $0 \to 0.98 \to 0$ grid-gap excursion at $(K, m, B) = (2, 8, 24)$ comes entirely from the upper endpoint. Exact enumeration of all 807 counts confirms that all 32 executable plans and lower values are unchanged. At $Q = 64$ one optimistic

key combines a first-coordinate minimum from one plan with a second-coordinate minimum from another. No single plan realizes both. Appendix H.8 gives those count witnesses and independently reconstructs the envelope. This is a valid but looser relaxation, not a worse deployed policy. Keeping the running intersection removes the excursion from the stopping procedure.

7 SCOPE, RELATED WORK, AND CONCLUSION

Restart schedules and algorithm selection precede this work (Luby et al., 1993; Gomes et al., 1998; Rice, 1976; Gomes & Selman, 2001). Dynamic restart policies already use updated beliefs about run time, including dependent runs (Kautz et al., 2002; Ruan et al., 2002); Streeter (2019) studies an expected-completion-time objective. Metareasoning evaluates computations by their effect on subsequent decisions (Russell & Wefald, 1991; Hay et al., 2012; Callaway et al., 2018; De Sabbata et al., 2025). These are closer conceptual comparisons than a generic claim about longer language-model reasoning. Our hard-budget reduction has a different interface, but that difference is not by itself a claim of a new general restart theory. The POMDP relationship is explicit in Section 3; temporal abstraction also has an established framework (Sutton et al., 1999).

Language-model tree search, feedback, and backtracking provide potential uses for such interfaces, not empirical validation of them (Yao et al., 2023; Hao et al., 2023; Zhou et al., 2024; Madaan et al., 2023; Shinn et al., 2023; Gou et al., 2024; Uscidda et al., 2025; Chitale et al., 2026). Process supervision and fallible verification are separate problems (Cobbe et al., 2021; Uesato et al., 2022; Lightman et al., 2024; Rohatgi et al., 2026). In particular, an imperfect-verifier guarantee cannot be inferred from our sound-checker model. We report no LLM benchmark, learned latent representation, or competitive SAT-solver result.

Validity can conflict with tractability. A small *global* specification family may govern many actions, as in the decoder. By contrast, m independent persistent usable/unusable bits yield $K = 2^m$ sufficient states. The explicit grid then becomes doubly exponential in m; α pruning may help particular instances but has no universal escape from the dimensional obstruction. Factorization, abstraction, or restricting the reachable beliefs could help, but requires additional algorithms and error guarantees. The dimension lower bound is stated separately from the grid upper bound; an additional two-state $\Omega(\epsilon^{-1/2})$ example shows precision can also force more plans, without claiming either lower bound matches the grid rate (Appendix G).

Conclusion. For formally checkable search with a supplied small model, stationary failure blocks can be separated from decisions that create information. Their exact-cost α-vector envelopes compose through refreshes and support executable lower and optimistic upper planning values. Standard pruning is often markedly better than the grid; simple bounds and greedy policies remain strong alternatives. When declared model rows can be sampled, simultaneous intervals extend a fitted-model bracket to an environment-level guarantee. The useful result is this explicit interface between search structure, established POMDP approximation, and measurable model uncertainty—not an unrestricted prescription to reuse, restart, or think longer.

AI USE STATEMENT

In this work, we used generative AI tools to draft sections of the paper and polish writing. Additionally, we used generative AI tools for editing code. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. URL https://web.stanford.edu/~boyd/cvxbook/.

Frederick Callaway, Sayan Gul, Paul M. Krueger, Thomas L. Griffiths, and Falk Lieder. Learning to select computations. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence*, 2018. URL https://arxiv.org/abs/1711.06892.

Anthony R. Cassandra. pomdp-solve: Command Line Options, Version 5.0 and Earlier. Official software documentation, n.d. URL https://www.pomdp.org/code/pomdp-solve-options-5.0.html. Options enum_purge, epsilon_prune, and terminal_values; accessed September 2026.

Anthony R. Cassandra, Michael L. Littman, and Nevin Lianwen Zhang. Incremental pruning: A simple, fast, exact method for partially observable markov decision processes. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pp. 54–61, 1997. URL https://arxiv.org/abs/1302.1525.

Rajiv Shailesh Chitale, Rahul Madhavan, Taneesh Gupta, Deepanway Ghosal, and Aravindan Raghuveer. Test-Time Scaling via Error Localization. *arXiv preprint arXiv:2607.21453*, 2026. URL https://arxiv.org/abs/2607.21453.

C. J. Clopper and E. S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413, 1934. doi: 10.1093/biomet/26.4.404. URL https://doi.org/10.1093/biomet/26.4.404.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*, 2021. URL https://arxiv.org/abs/2110.14168.

C. Nicolò De Sabbata, Theodore R. Sumers, Badr AlKhamissi, Antoine Bosselut, and Thomas L. Griffiths. Rational Metareasoning for Large Language Models. *arXiv preprint arXiv:2410.05563*, 2025. URL https://arxiv.org/abs/2410.05563v3. Version 3, revised June 2025.

Norman Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite markov decision processes. In *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence*, pp. 162–169, 2004. URL https://arxiv.org/abs/1207.4114. Proceedings paper; author-deposited version available on arXiv.

Carla P. Gomes and Bart Selman. Algorithm portfolios. *Artificial Intelligence*, 126(1–2):43–62, 2001. doi: 10.1016/S0004-3702(00)00081-3. URL https://doi.org/10.1016/S0004-3702(00)00081-3.

Carla P. Gomes, Bart Selman, and Henry Kautz. Boosting Combinatorial Search through Randomization. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pp. 431–437, 1998. URL https://cdn.aaai.org/AAAI/1998/AAAI98-061.pdf.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/notes/edits/attachment?id=2obldDZSzy&name=pdf.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with Language Model is Planning with World Model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 8154–8173. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.507. URL https://aclanthology.org/2023.emnlp-main.507/.

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*, 2012. URL https://arxiv.org/abs/1207.5879.

Henry Kautz, Eric Horvitz, Yongshao Ruan, Carla Gomes, and Bart Selman. Dynamic restart policies. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pp. 674–681, 2002. URL https://www.microsoft.com/en-us/research/publication/dynamic-restart-policies/.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison 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. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

William S. Lovejoy. Computationally feasible bounds for partially observed markov decision processes. *Operations Research*, 39(1):162–175, 1991. doi: 10.1287/opre.39.1.162. URL https://doi.org/10.1287/opre.39.1.162.

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. URL https://people.eecs.berkeley.edu/~sinclair/vegas.pdf.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative Refinement with Self-Feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Christos H. Papadimitriou and Mihalis Yannakakis. On the Approximability of Trade-offs and Optimal Access of Web Sources. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pp. 86–92, 2000. doi: 10.1109/SFCS.2000.892068. URL https://doi.org/10.1109/SFCS.2000.892068.

Joelle Pineau, Geoff Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pp. 1025–1032, 2003. URL https://www.ijcai.org/proceedings/2003.

Joelle Pineau, Geoffrey J. Gordon, and Sebastian Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, 27:335–380, 2006. doi: 10.1613/jair.2078. URL https://doi.org/10.1613/jair.2078.

John R. Rice. The algorithm selection problem. In *Advances in Computers*, volume 15, pp. 65–118. Academic Press, 1976. doi: 10.1016/S0065-2458(08)60520-3. URL https://doi.org/10.1016/S0065-2458(08)60520-3.

Dhruv Rohatgi, Abhishek Shetty, Donya Saless, Yuchen Li, Ankur Moitra, Andrej Risteski, and Dylan J. Foster. Taming Imperfect Process Verifiers: A Sampling Perspective on Backtracking. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.03149.

Yongshao Ruan, Eric Horvitz, and Henry Kautz. Restart policies with dependence among runs: A dynamic programming approach. In *Principles and Practice of Constraint Programming*, 2002. URL https://www.erichorvitz.com/dprestart.htm.

Stuart Russell and Eric Wefald. Principles of metareasoning. *Artificial Intelligence*, 49(1–3): 361–395, 1991. doi: 10.1016/0004-3702(91)90015-C. URL https://doi.org/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, pp. 8634–8652, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Richard D. Smallwood and Edward J. Sondik. The Optimal Control of Partially Observable Markov Processes over a Finite Horizon. *Operations Research*, 21(5):1071–1088, 1973. doi: 10.1287/opre.21.5.1071. URL https://doi.org/10.1287/opre.21.5.1071.

Matthew Streeter. Bayes Optimal Early Stopping Policies for Black-Box Optimization. *arXiv preprint arXiv:1902.08285*, 2019. URL https://arxiv.org/abs/1902.08285.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211, 1999. doi: 10.1016/S0004-3702(99)00052-1. URL https://www.sciencedirect.com/science/article/pii/S0004370299000521.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022. URL https://arxiv.org/abs/2211.14275.

Theo Uscidda, Matthew Trager, Michael Kleinman, Aditya Chattopadhyay, Wei Xia, and Stefano Soatto. LATTS: Locally Adaptive Test-Time Scaling. *arXiv preprint arXiv:2509.20368*, 2025. URL https://arxiv.org/abs/2509.20368.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 11809–11822, 2023. URL https://proceedings.neurips.cc/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract.html.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language Agent Tree Search Unifies Reasoning, Acting, and Planning in Language Models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62138–62160, 2024. URL https://proceedings.mlr.press/v235/zhou24r.html.

A POLICY REDUCTION: COMPLETE PROOFS AND BOUNDARIES

A.1 FAILURE-ONLY ALLOCATION

Condition on a controller's random seed; its policy is now deterministic. Delete terminal-success leaves. Every remaining completion observation is rejection, so the remaining tree is one path. Let n_d count action d on this path. Positive costs and the hard budget ensure $c^\top n \le B$. Conditional on hidden state ℓ, all its attempts fail with probability $f_n(\ell) = \prod_d (1 - s_d(\ell))^{n_d}$. Executing the recorded counts in a fixed order, stopping on success, is feasible and has that same complete-failure probability. Hence it has the same statewise success. Conversely any affordable count vector is an admissible policy. Averaging over controller seeds is a convex combination of deterministic success probabilities and cannot exceed their maximum. This proves Theorem 1. Bayesian updating after rejection is compatible with the result: it selects the next action on this one recorded path.

A.2 DIAGNOSTIC FRONTLOADING

Truncate a deterministic policy at termination or its first refresh, and delete all earlier success leaves. Completion nodes are unary; diagnostic nodes retain their outcome branches. Contract completion nodes. For each diagnostic leaf γ, store all contracted completion counts $n(\gamma)$ and the boundary action. The contracted tree asks the original diagnostics in their original diagnostic-outcome order, then executes its leaf's counts before the boundary action. It does not pretend to have observed unperformed failures. Its decisions are a compiled finite tree, not online posterior inference from fictitious outcomes.

Let $A_\gamma(\ell)$ be the product of diagnostic likelihoods on that path. In the original policy, the probability of reaching this leaf without success is $A_\gamma(\ell) f_{n(\gamma)}(\ell)$. The transformed policy has the same factors in a different order. Conditional independence and unchanged laws are exactly the assumptions needed. Summing over all nonsuccess leaves gives equal failure probability, so success agrees for each ℓ.

For feasibility, every original complete nonsuccess path has total cost at most B. Reordering positive-cost actions cannot make a prefix cost exceed that path's total. The transformed diagnostic prefix, count block, and boundary action are therefore affordable. Diagnostic identities, outcomes, and quota use are unchanged. All diagnostics are available at phase entry; no availability constraint is crossed. With no refresh this proves Theorem 2. A randomized policy follows by conditioning and averaging over its seed. Reordering can increase expected expenditure, since an original early success could have avoided a diagnostic; the theorem only preserves worst-case feasibility and success before the deadline.

A.3 THE UNSUCCESSFUL EXIT LAW AND COMPOSITION

For a refresh h after leaf γ, the joint mass of nonsuccess exit to output z and hidden state ℓ' from initial ℓ is

$$A_\gamma(\ell) f_{n(\gamma)}(\ell) T_{h,z}(\ell, \ell'; x).$$

The two policies give exactly the same product. Remaining budget is

$$B - \sum_{j \text{ on } \gamma} a_j(x) - c(x)^\top n(\gamma) - a_h(x),$$

and remaining quotas are determined by the same diagnostic path and boundary action. The new bank $x_{h,z}$ and every exposed diagnostic identity and result also agree. Stopping can be encoded as an identity exit with no future value. Integrating over initial state and paths preserves the entire joint nonsuccess law; its complement is phase success. The allocation of early success probability to specific actions need not agree.

For zero refresh quota the result is the diagnostic theorem. Inductively, transform the first phase and attach an equivalent continuation at every observed exit with its unchanged posterior, bank, and remaining resources. The identical joint exit distribution gives identical continuation value. This proves Theorem 3 by induction on refresh quota, or on budget. It also establishes that the diagnostic/block/refresh recursion is complete, rather than merely a feasible subclass of policies. The recursion uses unnormalized masses; impossible outcomes have zero mass and need no arbitrary

posterior definition. The value is positively homogeneous, between zero and $\|u\|_1$, and zero when no useful action is affordable.

A.4 Boundaries and a possibility result

The assumptions are sufficient, not necessary. Some richer-feedback instances admit the same optimum by coincidence. Conversely, new information can make global frontloading strictly worse. A cost-one refresh creates a fair bit Z and two cost-two completions, successful exactly when their index equals Z. A cost-one diagnostic, available only after refresh, reveals Z. At budget four, refresh–diagnose–complete succeeds with probability one. A policy that prohibits diagnostics after refresh can afford only one completion and succeeds with probability at most one-half. Retaining an old always-failing root makes this a growing-bank example. This construction establishes *possibility*, not prevalence. The random checked-refresh study often has negligible loss from global frontloading.

A free counterexample, a retained fragment from a failed completion, a changing proposal distribution, or a diagnostic that becomes available after generation violates a plain-completion phase description unless modeled as a refresh. Correlated persistent outcomes can be represented by a sufficient latent state, but adding those states can make explicit planning impractical. No part of the proof validates an application-specific hidden-state model automatically.

B Cost-sliced α pruning and refresh composition

This section uses standard value-envelope dominance (Smallwood & Sondik, 1973; Cassandra et al., 1997; Cassandra, n.d.). It proves its use as an exact-cost block oracle; α-vectors, linear-program pruning, and their error principle are not proposed as new.

B.1 Budget augmentation and a fair size comparison

Adjoin observed bank, remaining budget, and quotas to a POMDP's state. Positive integer costs make the horizon finite. Success has value one; exhausting the budget has value zero. For each observed component, a policy's value is a K-vector over the remaining hidden state. Observable budget means beliefs never mix different budget coordinates. Therefore the correct componentwise-grid comparison is at most $(B+1)(Q+1)^K$ records per bank, before accounting for quotas and refresh beliefs. Using $(Q+1)^{K(B+1)}$ as a supposedly necessary POMDP baseline would ignore observability and be misleading. Exact-cost storage is one way to precompute an open-ended block independent of its eventual continuation, but is expressible in this conventional state-augmented formulation.

A generic point-based approximation retains value vectors near sampled beliefs. Its coverage and backup analysis, rather than a count of grid records in latent coordinates, controls its error (Pineau et al., 2003; 2006). We compare a simple full-simplex LP-pruning implementation, not a complete optimized point-based or incremental-pruning package. Pruning complexity can still be exponential; no uniform polynomial record bound is asserted for the LP implementation.

B.2 Envelope loss is enough; a single nearby vector is unnecessary

For finite executable sets $A, \widehat{A} \subset [0,1]^K$, suppose

$$\sup_{v \geq 0, \|v\|_1 = 1} \{\max_{\alpha \in A} v^\top \alpha - \max_{\beta \in \widehat{A}} v^\top \beta\} \leq e.$$

Homogeneity gives the same bound times $\|v\|_1$ without normalization. For a fixed continuation $h \in [0,1]^K$, the block value is

$$u^\top h + [u \odot (1-h)]^\top \alpha.$$

Applying the envelope inequality to $v = u \odot (1-h)$ loses at most $e\|u\|_1$. A different representative may be chosen for each u, h. For example, $(.45, .45)$ is exactly envelope-dominated by the set $\{(.9, 0), (0, .9)\}$, although neither vector individually componentwise dominates it. Requiring one sup-norm-near vector would be unnecessarily strong.

The argument still applies when an exact policy selects its later actions using the posterior after the original block: fix that policy as a mapping of future public histories. At the replacement block's

exit it remains feasible because exact expenditure, bank, and quotas agree. Its success vector h is well-defined even though it may no longer be optimal under the replacement posterior. The replacement planner can always select it and can only improve by reoptimizing. This is the role of exact-cost unsuccessful exits; no assertion about closeness of normalized posteriors is needed.

B.3 Constructing the library by positive-cost backups

Let A_t contain every exact count-plan success vector at cost t. The zero-cost set contains only zero. For each action d and retained parent $\beta \in \widehat{A}_{t-c_d}$, produce

$$\mathcal{B}_d\beta = s_d + (1 - s_d) \odot \beta.$$

Retain its integer counts as an executable witness. Equal counts may be deduplicated. Let the finite candidate pool be C_t. If the parent envelope losses are e_{t-c_d}, then for nonnegative v the maximum over A_t exceeds the maximum over C_t by at most

$$\max_{d:c_d\le t} e_{t-c_d}\,\|(1 - s_d) \odot v\|_1 \le \big(\max_{d:c_d\le t} e_{t-c_d}\big)\|v\|_1,$$

with the maximum restricted to reachable parent costs. Constants $v^\top s_d$ cancel in each comparison.

Process candidates in a deterministic order. Omit α only when

$$\max_{w\ge 0,\mathbf{1}^\top w=1}\left[w^\top\alpha - \max_{\beta\in R} w^\top\beta\right] \le \rho, \tag{8}$$

where R is the currently retained set. Later retention only enlarges R, so every omitted candidate is dominated by the final retained set with its recorded loss. If d_t is the maximum nonnegative omitted loss, the full candidate envelope is within $d_t \le \rho$. Hence

$$e_0 = 0, \qquad e_t = \max_{d:t-c_d \text{ reachable}} e_{t-c_d} + d_t \le \lfloor t/c_* \rfloor \rho.$$

Keeping the computed d_t often gives a smaller certificate than replacing it by ρ everywhere. Every retained vector is still an actual count plan. The only approximation is omission of envelope-negligible candidates.

A dual witness for numerical pruning. The LP in (8) is a finite zero-sum game. Its value is also

$$\min_{\lambda\ge 0,\mathbf{1}^\top\lambda=1}\max_{\ell}\left[\alpha_\ell - \sum_{\beta\in R}\lambda_\beta\beta_\ell\right].$$

Any feasible mixture λ supplies an upper bound on the candidate's advantage, whether or not optimization has converged. The code extracts nonnegative dual weights, normalizes them, and evaluates this feasible witness. A componentwise witness is the special case selecting one retained vector. This avoids treating a primal feasible LP value as an upper certificate. Floating-point implementations add a 10^{-10} local safety allowance; all reported numerical comparisons use explicitly recorded tolerances. These are not formal interval-arithmetic proofs of floating-point execution.

B.4 Lower and upper phase recursions

The lower recursion uses only retained count witnesses and exact model kernels. Its value L is therefore achievable. For the upper recursion, use the same count candidates, add $e_t\|u\|_1$ to a block at cost t, use upper future values at refresh exits, and cap the action's value by $\|u\|_1$. Diagnostics use their ordinary mass-weighted backups. We prove $V^* \le U$ by budget induction. For an exact optimal block and its fixed optimal subsequent policy, envelope loss gives a retained block with exact future value at least $V^*_{\text{block}} - e_t\|u\|_1$. Induction makes that block's upper future value no smaller. Adding the allowance proves optimism. A diagnostic's exact children are bounded by the upper children by induction. Taking maxima and capping by a valid universal upper bound preserve the inequality.

For the computed gap, compare an upper-optimal action to the same feasible action in the lower recursion. A block incurs allowance at most $\lfloor t/c_* \rfloor \rho\|u\|_1$. If it refreshes, the sum of nonsuccess exit masses is at most $\|u\|_1$ and the remaining completion allowance is at most $\lfloor (b - t - a_h)/c_* \rfloor \rho$. Thus the total is at most $\lfloor b/c_* \rfloor \rho\|u\|_1$. At a diagnostic, outcome masses sum to the parent mass and remaining budget decreases. Stop and terminal count blocks satisfy the same inequality. Maximizing lower actions can only tighten the gap. With normalized initial mass this proves Theorem 5.

Why the implementation adds scalar mass, not clipped vectors. An envelope allowance need not be attributable to one componentwise optimistic success vector. Clipping $\beta + e_t\mathbf{1}$ at one before a later continuation can reduce the promised allowance in state-dependent ways. The proof instead adds the scalar $e_t\|u\|_1$ after evaluating the whole block and continuation, then clips the total by the valid bound $\|u\|_1$. That is the recursion tested against the independent primitive planner.

C ROUNDED-GRID CONSTRUCTION AND ITS FINITE ERROR BOUND

This is the rounded-grid reference oracle. Its success vectors are α-vectors; vector reuse across beliefs is standard POMDP structure. The proof below supplies exact-cost and phase-error bookkeeping for this oracle, not a new principle of value representation. Throughout, the model kernels are known, costs are positive integers, and all bank states have at most K hidden states. Padding smaller state spaces with unused coordinates is harmless. Write c_* for the minimum completion cost over the bank states under consideration.

C.1 CONSTRUCTION AT EXACT COST

For one bank, write $a_{d\ell} = 1 - s_d(\ell)$. Fix a positive integer Q. A library record is (t, k, n, f), where t is exact spent cost, $k \in \{0, \dots, Q\}^K$ is a rounded vector, n is a feasible count vector, and $f_\ell = \prod_d a_{d\ell}^{n_d}$ is its *actual* complete-failure probability. Begin with $(0, Q\mathbf{1}, 0, \mathbf{1})$. Appending action d gives

$$t' = t + c_d, \qquad k'_\ell = \lceil a_{d\ell} k_\ell \rceil, \qquad n' = n + e_d, \qquad f' = f \odot a_d.$$

Retain one record per (t', k') and discard the others. Process costs in increasing order through B. A rounded vector can depend on the order used to reach it; no order-independence of rounding is assumed. Nevertheless, every count plan has at least one represented rounded path.

Rounding invariant. After a sequence of j completions, its exact vector f and rounded vector k satisfy, coordinatewise,

$$0 \le k/Q - f \le (j/Q)\mathbf{1}. \tag{9}$$

Initially the difference is zero. A subsequent multiplication by $a_d \in [0,1]^K$ cannot increase the largest difference, and upward rounding adds less than $1/Q$ in each coordinate. This proves the invariant by induction. It applies separately to the actual sequence underlying every retained record.

Take any target plan of exact cost t and any ordering of its counts. Its rounded path remains reachable: if an intermediate record is replaced, the rounded key and exact cost are unchanged, so every subsequent action of the target sequence remains affordable and has the same rounded transition. At the final key, both the target vector f and the retained vector $\widetilde{f}$ lie in

$$[\, k/Q - \lfloor t/c_* \rfloor \mathbf{1}/Q,\ k/Q \,].$$

Consequently,

$$\|f - \widetilde{f}\|_\infty \le \lfloor t/c_* \rfloor / Q. \tag{10}$$

The representative spends *exactly* t, not merely at most t. This is essential when its unsuccessful exit is followed by a budget-sensitive continuation.

Size and work. There are at most $(B+1)(Q+1)^K$ keys, with at most m candidate extensions from each reachable key. Thus construction needs at most $m(B+1)(Q+1)^K$ candidate transitions, each with K coordinate operations. Predecessor pointers suffice to recover the count vector, so the mathematical representation need not store m counts at every key. The supplied implementation does store those counts for convenient checking; its record storage is $O((m+K)(B+1)(Q+1)^K)$. Exact arithmetic bit costs and the number of observed banks are not suppressed in a claim of an overall polynomial-time controller: no such general claim is made.

C.2 WHY THE LIBRARY IS VALID FOR EVERY CONTINUATION

Fix the current unnormalized hidden-state mass u, an exact cost t, and a chosen refresh or stop. For a fixed feasible continuation policy, let $h_\ell \in [0,1]$ be its probability of eventual success, conditional on

surviving the count block in state ℓ; h includes the refresh outcome and transition. The block and its continuation have value

$$G_h(u, f) = \sum_{\ell} u_\ell[1 - f_\ell + f_\ell h_\ell]. \tag{11}$$

This function is decreasing in each f_ℓ, and

$$|G_h(u, f) - G_h(u, g)| \le \|u\|_1 \|f - g\|_\infty.$$

The same statements hold after maximizing over continuation policies. The set of permitted continuations is the same because exact spent cost, observed bank, diagnostic quota and refresh quota are unchanged. A representation that preserved only $u^\top f$ at one prior would not have this property.

For each retained key define

$$\underline{f}(t, k) = \max\{0, k/Q - \lfloor t/c_* \rfloor \mathbf{1}/Q\}. \tag{12}$$

Use the bank-specific minimum completion cost instead of c_* to obtain a tighter valid envelope. For the target plan and its representative,

$$\underline{f} \le f, \widetilde{f} \le k/Q, \qquad 0 \le \widetilde{f} - \underline{f} \le \lfloor t/c_* \rfloor \mathbf{1}/Q.$$

The implementation tightens this envelope without adding grid coordinates. For each key, it propagates the coordinatewise minimum of actual failure vectors over all ordered paths reaching that key. At an extension, multiply the parent's minimum by a_d; at a merge, take coordinatewise minima. Multiplication by nonnegative a_d commutes with a coordinatewise minimum, so induction shows these are the exact per-coordinate minima of that path group (in exact arithmetic). Every target path and retained representative lies above them, and the rounding invariant bounds their distance from the representative by $\lfloor t/c_* \rfloor/Q$. They can therefore replace (12) throughout the proof. The minima need not belong to one feasible plan; only the lower bound is executed.

The phase Bellman recursion evaluated with actual representative vectors is an executable lower bound L. The same recursion evaluated with $\underline{f}$ is an optimistic upper bound U: every exact block has an optimistic record at the same cost that dominates it in (11). Induction over remaining budget, including diagnostics and refresh transitions, therefore gives $L \le V^* \le U$.

A global error budget, not one penalty per branch. We prove the stronger statement

$$U(b, x, u, k, r) - L(b, x, u, k, r) \le \|u\|_1 \lfloor b/c_* \rfloor/Q.$$

At a diagnostic, the outcome masses sum to $\|u\|_1$, and every child has strictly smaller budget. At a block of exact cost t, changing its optimistic vector to its executable vector costs at most $\|u\|_1 \lfloor t/c_* \rfloor/Q$ by (11). Replacing the upper continuation by the lower continuation costs at most its nonsuccess mass times $\lfloor (b - t - a_h)/c_* \rfloor/Q$. This mass is at most $\|u\|_1$, and

$$\lfloor t/c_* \rfloor + \lfloor (b - t - a_h)/c_* \rfloor \le \lfloor b/c_* \rfloor.$$

Stopping has no continuation loss. Comparing the upper-optimal action with the same feasible lower action establishes the induction; maximizing the lower actions cannot worsen it. Values are also between zero and $\|u\|_1$. For a normalized initial prior this proves Theorem 4, including its bound independent of the number of diagnostic branches or refresh phases.

Refinement and modeling error. Increasing Q need not make the particular retained representatives nested. An implementation can retain the best executable lower value and the smallest upper value across refinements; both remain valid. Choosing $Q \ge \lfloor B/c_* \rfloor/\epsilon$ suffices for gap ϵ. These certificates concern the supplied finite-state model. Applying the planner to estimated rates does not certify performance under the unknown true model; model-learning error is a separate issue.

D From sampled completion rows to an environment bracket

The statistical claim here is separate from generalization across unseen tasks and from estimating a hidden-state abstraction. The support, legal actions, initial prior, and diagnostic/refresh kernels are supplied. Only success probabilities of a finite set of queryable labeled completion rows are unknown.

D.1 SIMULTANEOUS ROW INTERVALS

For a declared row i, calibration produces $X_i \sim \mathrm{Binomial}(R_i, s_i)$ from independent invocations conditional on its label. Fix a finite collection $\mathcal{F}$ of row/sample-size pairs, of size M. For each pair let $\gamma = \delta/(2M)$. The interval is

$$l_i = \begin{cases} 0 & X_i = 0, \\ \mathrm{Beta}^{-1}(\gamma; X_i, R_i - X_i + 1) & X_i > 0, \end{cases}$$

$$u_i = \begin{cases} 1 & X_i = R_i, \\ \mathrm{Beta}^{-1}(1-\gamma; X_i + 1, R_i - X_i) & X_i < R_i. \end{cases}$$

Clopper–Pearson inversion (Clopper & Pearson, 1934) bounds each one-sided noncoverage probability by γ. A union bound gives simultaneous coverage at least $1 - \delta$ over $\mathcal{F}$. Independence across pairs is not required: nested rollout prefixes and competing methods may reuse samples. Conditional independence within each sampled row is required. Known duplicate rows across banks share one interval rather than being counted as distinct observations.

A data-selected planner, prior query, or recorded grid resolution remains valid on the same simultaneous event. Adaptive stopping at arbitrary unlisted sample sizes is not licensed; it would need a confidence sequence or a separately allocated error budget. The experiment lists five sample sizes in advance and includes all of them in $M = 28 \times 5 = 140$.

D.2 MONOTONE COMPLETION KERNELS

Fix any feasible public-history policy, including all random seeds. Write its value conditional on a hidden state and history. At a completion node with success rate s_i, it equals $s_i + (1 - s_i)H$, where $0 \le H \le 1$ is the failure continuation value. This is nondecreasing in s_i. By induction on remaining budget, replacing future completion rates by larger ones also cannot decrease H. A diagnostic or refresh takes a nonnegative weighted sum of successor values with identical, known weights, so preserves monotonicity. Stopping is constant. Thus every fixed policy's success is nondecreasing under a simultaneous componentwise increase of completion rates.

On the row-coverage event, every policy satisfies $V^-(\pi) \le V_{\mathrm{true}}(\pi) \le V^+(\pi)$. Let π_- be the executable lower-endpoint policy and $L^- = V^-(\pi_-)$. Then $L^- \le V_{\mathrm{true}}(\pi_-) \le V^*_{\mathrm{true}}$. Conversely, any upper bound for all upper-endpoint policies also bounds the true optimum. This gives (7). The proof holds for policies chosen from calibration data: after conditioning on a dataset in the simultaneous event, the fixed-policy monotonicity statement holds for all policies. The lower endpoint need not be the exact lower-model optimum; an executable approximate policy is enough. Likewise the upper planner may be a valid grid or α relaxation.

D.3 ACQUIRED TV ERRORS AND WHAT THEY COVER

A plain completion has one success output and one failure output leaving the same state. The TV distance between two such kernels is exactly the difference in their Bernoulli success rates. For an estimated row $\widehat{s}_i$, simultaneous intervals therefore give

$$|s_i - \widehat{s}_i| \le \max\{\widehat{s}_i - l_i, u_i - \widehat{s}_i\}.$$

Maximizing over the declared modes yields η_d for each proposal menu. Known diagnostic and refresh kernels have error zero. The coupling result in Appendix E then transfers the nominal bracket. Taking a maximum over a union of potentially affordable action types, while ignoring quotas, is conservative. Completion-only endpoint monotonicity is tighter here because it avoids charging every possible kernel mismatch as total loss of value.

Non-identifiability without an observed support or structural restriction. No finite calibration procedure observing only a subset of fine states can infer a uniform error bound on all unobserved states without further assumptions. Two environments can agree exactly on every queried row and differ arbitrarily on an unqueried row. Their calibration data distributions are identical, while a policy reaching that row can have different success. Similarly, averaging an always-usable and an always-unusable persistent state produces the correct initial mean but the wrong repeated-failure law. The row theorem does not bridge either gap. The supplied decoder family avoids it by enumeration of modes and labeled sampling access, not by learning an abstraction from a few traces.

E SMALL SUFFICIENT MODELS AND CONTROLLED AGGREGATION ERROR

A small hidden-state model is an assumption to justify, not an outcome of rounded allocation. Exact state abstraction and quantitative state aggregation are established topics (Ferns et al., 2004). We give a budget-sensitive application of the standard coupling argument to explain precisely how an independently justified abstraction error enlarges our planning certificate. No abstraction-learning algorithm is claimed.

Let the actual sufficient state be z in a finite space, and let $\phi_x(z) \in \{1, \dots, K\}$ be its proposed aggregate at observed bank x. Legal actions, costs, and observed bank updates must agree between models. For action a, push the fine kernel forward to the pair *(public output, next aggregate)*; terminal success is a public absorbing output. Suppose for *every* fine state z this distribution is within total variation η_a of the abstract kernel at $\phi_x(z)$. The bound is uniform over reachable banks and permitted history-dependent invocations. It cannot be replaced by average prediction error under one calibration distribution. The pushforward initial prior is within η_0 of the supplied abstract prior.

For positive integer action costs, put $D_0 = 0$ and

$$D_b = \max\left\{0, \max_{a: c_a \le b} [\eta_a + (1-\eta_a) D_{b-c_a}]\right\}, \qquad E_B = \eta_0 + (1-\eta_0) D_B. \tag{13}$$

Using the union of action types and ignoring quotas is conservative. A quota-aware recursion may tighten it. Equivalently, D_B is the largest $1 - \prod_t (1-\eta_{a_t})$ over affordable action sequences. The elementary bound is $E_B \le \min\{1, \eta_0 + \max_{c^\top n \le B} \sum_a n_a \eta_a\}$.

Proposition 7 (Transporting a supplied model certificate)**.** *Under the uniform aggregate-kernel condition, every shared observation-history policy π satisfies*

$$|V_{\text{fine}}(\pi) - V_{\text{abstract}}(\pi)| \le E_B.$$

If an executable abstract policy π_L has value L and all abstract policies have value at most U, then

$$V^*_{\text{fine}} \in [\max\{0, L - E_B\}, \min\{1, U + E_B\}], \qquad V^*_{\text{fine}} - V_{\text{fine}}(\pi_L) \le \min\{1, U - L + 2E_B\}. \tag{14}$$

The comparator receives the same public observations; it does not observe the fine state for free.

Proof. Couple the initial abstract state to the image of the fine state, with mismatch probability at most η_0. Give both systems the same controller random seed. While public histories and aggregate states match, the policy chooses the same legal action and pays the same cost. Maximally couple its abstract output/next-state pair to the corresponding fine pushforward. The probability of mismatch at this step is at most η_a. Conditional on a matched pair, sample the fine next state from its regular conditional distribution within that pair. This preserves both process marginals; it does not assume that fine states within an aggregate have identical transitions.

Until the first mismatch, all public histories, stopping decisions, budgets and success indicators coincide. Conditional on any matched history, backward induction on remaining budget bounds the chance of any later mismatch by D_b: failure of the current coupling costs at most η_a, while success leaves at most D_{b-c_a} additional risk. Initial coupling gives E_B. Different terminal success indicators require a mismatch, proving the uniform fixed-policy bound. This argument works for arbitrary full-history policies; it never divides by a small observation probability or asserts a bound on normalized posterior error.

For the lower endpoint, lift π_L to the fine process, giving at least $L - E_B$ and therefore the same lower bound on the fine optimum. For the upper endpoint, apply the uniform bound to a fine-optimal observation-history policy, whose abstract value cannot exceed U. Subtract the lifted policy's guarantee to obtain regret at most $U - L + 2E_B$, then cap all probability bounds by one. □

Corollary 8 (Same-state model perturbation)**.** *With the identity map, a common per-step error η, and minimum action cost $c_{\min}$, $E_B \le \min\{1, \eta_0 + \lfloor B/c_{\min} \rfloor \eta\}$. The familiar twice-model-error regret guarantee follows when $L = U$.*

This recovers the same-state result without duplicating its proof. It covers supplied bounds on rate error as well as aggregation only when their *joint* effect meets the kernel condition. A model-fitted library bracket is not a true-environment bracket unless such bounds are independently available.

Why matching averaged rates is not enough. Suppose a hidden persistent state is equally likely always successful or always unsuccessful. Merging both states gives the correct initial mean success .5. The true probability of success in two retries is .5, whereas treating the merged .5 rate as independent predicts .75. Conditioning on failures changes the fine-state mixture. Exact lumpability gives $E_B = 0$; arbitrary averaging does not. Our error check uses explicit uniform row bounds, rather than using zero initial average error as a certificate.

Empirical check. Forty constructed model pairs use two abstract states and six fine states, with three persistent nuisance refinements per abstract state, two completion actions, one diagnostic, and a stochastic refresh to a second bank. Declared perturbation levels are $0, 2 \times 10^{-3}, .01, .04$ across ten seeds. Every pushed-forward row error is checked. We compare an exact fine-model primitive planner with a compact-model library policy, evaluated using separate abstract and fine probability masses. All transported brackets and regret bounds hold numerically. These finite checks test the coupling implementation, not discovery of a realistic abstraction.

F Independent upper bounds and what they certify

This section concerns a fixed bank with no further diagnostic or refresh. It does not replace the phase certificate. All costs are positive integers, the budget is a nonnegative integer, and the prior is normalized. The comparison uses two inexpensive alternatives, neither claimed as a new optimization method.

F.1 State-observing integer allocation

For $a_{d\ell} = -\log(1 - s_d(\ell))$, define $A_b(\ell)$ as the largest accumulated log-failure reduction obtainable at integer budget b in state ℓ. The unbounded-knapsack recurrence is

$$A_b(\ell) = \max\{A_{b-1}(\ell), \max_{d:c_d \le b}[a_{d\ell} + A_{b-c_d}(\ell)]\}, \qquad A_0 = 0.$$

For interior probabilities this takes $O(mBK)$ arithmetic operations; success-one actions can instead be handled with a direct failure-product recurrence. The resulting perfect-information (PI) bound is

$$U_{\mathrm{PI}}(w) = \sum_{\ell} w_\ell [1 - \exp(-A_B(\ell))]. \tag{15}$$

It permits a different allocation in every hidden state, whereas the controller does not know that state. Precomputing the K statewise optima makes each new-prior query $O(K)$.

Proposition 9 (A hierarchy for terminal queries)**.** *For the coordinatewise-minimum optimistic envelope used by our library, exact arithmetic gives*

$$V^*(w) \le U_{\mathrm{lib}}(w) \le U_{\mathrm{PI}}(w).$$

Proof. The first inequality is Theorem 4. At any library key, each coordinatewise minimum is taken over a subset of feasible ordered paths. It is therefore no smaller than the unrestricted statewise minimum of failure. Multiplication by nonnegative prior weights and minimization over keys preserve that comparison. Subtracting from one reverses the inequality for success. This argument uses the propagated coordinatewise minima, not every possible coarser rounding envelope. □

Thus PI cannot tighten this particular library's terminal upper bound. Its relevance is its negligible cost: it can already suffice when the incumbent is near the value obtained when the state is known. For the phase check, we also compute a *full* state-observing Markov decision process (MDP) bound: the latent state is revealed initially and after every refresh. Completion and refresh actions are optimized by a cost/quota-indexed Bellman recursion. Diagnostics have no value once the sufficient state is known. This is distinct from applying (15) to only one bank.

F.2 Continuous counts with a checked first-order certificate

Assume $0 \le s_d(\ell) < 1$. With $a_{d\ell}$ as above, failure for real counts is

$$F(x) = \sum_{\ell} w_\ell \exp\left(-\sum_{d} a_{d\ell} x_d\right), \qquad x \ge 0, \quad c^\top x \le B. \tag{16}$$

Each summand is an exponential of an affine function; hence F is convex. The continuous optimum F^*_{cvx} is no larger than the integer failure optimum, so $1 - F^*_{\mathrm{cvx}}$ is an upper bound on integer success. This follows from elementary convex optimization (Boyd & Vandenberghe, 2004).

A numerical primal value must not be mistaken for that upper bound. For *any* point z, convexity gives a tangent lower bound. Minimizing its linear part over the nonnegative budget simplex yields

$$F^*_{\mathrm{cvx}} \ge b(z) := F(z) - \nabla F(z)^\top z + B \min\{0, \min_d \nabla_d F(z)/c_d\}. \tag{17}$$

Therefore $U_{\mathrm{cvx}}(z) = 1 - \max\{0, b(z)\}$ is a valid upper bound even if optimization stops early. For feasible z, $F(z) - \max\{0, b(z)\}$ reports the relaxation's optimization gap, not its integrality gap. The code optimizes normalized cost shares with sequential least squares programming (SLSQP), restores primal feasibility, and recomputes (17). It records convergence and residuals. A separate high-precision check recomputes the tangent at saved optimizer points. Floating-point padding and those checks are not interval-arithmetic or formal verification of the full program.

The library and continuous bounds are incomparable. Take two unit-cost actions at budget one, uniform prior, and success vectors $(.9, .1)$ and $(.1, .9)$. The integer optimum is .5; an equal fractional mixture succeeds with probability .7. A sufficiently fine library is exact and gives .5, tighter than the continuous relaxation. At $Q = 1$, both actions share one cost-one key and its coordinatewise-minimum envelope gives .9, which is looser than .7. Both effects are tested. A continuous relaxation is especially effective when an integer-feasible action mixture already attains its optimum. We report these cases rather than treating every tight library bracket as evidence that compilation was necessary.

What does not carry across a refresh. If the future policy is fixed with statewise success h_ℓ, the combined objective is $1 - \sum_\ell w_\ell(1 - h_\ell) f_x(\ell)$ at a fixed block cost. A weighted convex relaxation still applies. But an *optimized* diagnostic/refresh continuation is not one fixed h: its value is a maximum over future policies and the posterior depends on f_x. The maximum of these concave success objectives need not be concave. Thus a terminal-prior convex bound cannot simply be inserted as an exact-cost exit vector. This does not rule out sophisticated relaxation-based phase solvers; it states why this inexpensive scalar baseline and the reusable vector object solve different problems.

G Two nonmatching lower bounds for universal libraries

The following obstructions are separate from the grid upper bound. Neither proves that the resolution dependence of that construction is optimal.

G.1 Dependence on state dimension

Let $K = 2h$, with K completion actions of unit cost and budget h. Action i succeeds deterministically in hidden state i and never otherwise. A count vector succeeds exactly on the set of distinct actions it contains, a set of size at most h.

For every h-element subset S of $\{1, \dots, K\}$, consider the prior uniform on S. Its optimal success is one. A library with additive loss at most ϵ for every prior must therefore contain, for every S, some set T with $|S \cap T| \ge (1 - \epsilon)h$. A smaller T may be padded to size h when upper-bounding how many targets it covers. One fixed h-set T covers at most

$$\sum_{j=0}^{\lfloor \epsilon h \rfloor} \binom{h}{j}^2 \le \left(\sum_{j=0}^{\lfloor \epsilon h \rfloor} \binom{h}{j}\right)^2 \le \exp\{2hH(\epsilon)\}$$

targets, where $H(p) = -p \log p - (1-p)\log(1-p)$ and $0 \le \epsilon < 1/2$. The final inequality follows from the binomial theorem: for $j \le \epsilon h$, $\epsilon^j (1-\epsilon)^{h-j} \ge \exp[-hH(\epsilon)]$. At $\epsilon = 0$ the inequality follows directly. There are $\binom{K}{h} \ge 2^K/(K+1)$ target sets, because the central binomial coefficient is the largest of the $K+1$ coefficients. Hence the number M of library plans must satisfy

$$M \ge \frac{\exp\{K[\log 2 - H(\epsilon)]\}}{K+1}. \tag{18}$$

The conclusion is about one library that works for *all priors*. This construction is easy to solve for a single given prior. It is neither a lower bound on every prior-specific solver nor a claim that the precise upper-bound dependence on Q is optimal.

G.2 Dependence on precision with two states

For $0 < \epsilon \le 1/64$, let $M = 1 + \lfloor 1/(4\sqrt{\epsilon}) \rfloor$, $\theta_i = 4i\sqrt{\epsilon}$, $i = 0, \dots, M-1$, and define M unit-cost actions with success vectors

$$\alpha_i = \left(\tfrac{1}{4} + \tfrac{1}{2}\cos\theta_i,\ \tfrac{1}{4} + \tfrac{1}{2}\sin\theta_i\right).$$

All coordinates lie in $[0,1]$. Budget one permits a single action or stopping. Under prior $w_i = (\cos\theta_i, \sin\theta_i)/(\cos\theta_i + \sin\theta_i)$, action i is uniquely best. For $j \ne i$,

$$w_i^\top(\alpha_i - \alpha_j) = \frac{1 - \cos(\theta_i - \theta_j)}{2(\cos\theta_i + \sin\theta_i)} \ge \frac{(\theta_i - \theta_j)^2}{8\sqrt{2}} > \epsilon.$$

Here all angles lie in $[0,1]$, $1 - \cos x \ge x^2/4$ there, and distinct angles differ by at least $4\sqrt{\epsilon}$. Stopping is still worse. A universal executable subset with loss at most ϵ must retain every action. Thus even $K = 2, B = 1$ can require $\Omega(\epsilon^{-1/2})$ plans. Randomized mixtures of retained actions cannot improve their maximum linear value at w_i.

This is an elementary convex-envelope packing example, not a matching lower bound for the grid's ϵ^{-K} dependence. It complements the state-dimension obstruction and prevents presenting that obstruction as evidence that the particular grid is optimal. It also does not lower-bound a solver specialized to just one given prior.

H Experimental protocols and complete diagnostic checks

H.1 Computational protocols

The main text separates known-model approximation, fitted-model policy comparisons, and an acquired true-environment interval. Table 2 maps each mathematical claim to its model assumptions. Independent primitive and phase recursions check the planning values and executable policies.

Timings are measured elapsed times for the specified workloads, not universal runtime guarantees. Support evaluations have a 20-second time limit and a separate memory-allocation limit. Resource caps include construction and query work. Timing measurements are excluded from deterministic-value reproduction comparisons, never from the reported runtime tables. Different study families use separate fixed seed namespaces; intentional sample sharing is documented rather than counted as additional independent data.

Numerical assertions allow documented floating-point tolerances. The mathematical results are proved in exact arithmetic; the program is not formally verified or implemented entirely in interval arithmetic.

H.2 Terminal representations and the continuous comparator

The moderate family draws costs uniformly from $\{2, \dots, 7\}$ and rates $s_d(\ell) = .35\sqrt{c_d}U_{d\ell}$, with independent $U_{d\ell} \sim \mathrm{Uniform}(.005, .20)$. The generator seed is $910000 + 100m + B$; priors are symmetric Dirichlet draws with all concentration parameters equal to one. The main comparison bank has seed 913224, $m = 32, B = 24, K = 2$, and 2,424,107 feasible count vectors. The 32-prior representation comparisons use the same 32 priors for all methods. Each algorithm is told the same

supplied bank model. The grid's lower plan is evaluated from actual rates, never rounded rates. Its upper vector is a coordinatewise minimum over a rounded path class and is not executed. α pruning uses $\rho = .01/\lfloor B/c_* \rfloor$; retained vectors always have integer executable count witnesses.

The LP baseline is a straightforward cost-sliced implementation of full-simplex ε dominance, using a componentwise check before solving an LP. Candidate ordering is deterministic (decreasing unweighted success sum, then count tuple), not optimized per prior. It is not the external `pomdp-solve` program, a full point-based solver, or a new pruning method. The positive-cost backup and computed error recurrence in Appendix B are used for every retained set. Table 5 gives all completed $Q = 128$ comparisons. The $m = 32, B = 24, K = 3$ evaluation reaches its 25-second cap in the combined comparison; the incomplete case remains in the denominator of twelve comparisons.

Table 5: All completed moderate-bank comparisons at grid $Q = 128$. Gaps are maximum computed bracket widths over the same 32 priors, in percentage points. The α set has a one-point global per-backup error allowance. Values printed as zero are smaller than the displayed precision, not claims of formally exact floating-point equality. The twelfth combined evaluation is resource-limited.

(m, B, K)	Feasible counts	Grid records	α records	Grid gap	α gap
(8,12,2)	153	286	34	0.00	0.31
(8,24,2)	807	2,674	102	0.00	0.00
(16,12,2)	1,130	1,605	79	0.00	0.00
(16,24,2)	12,857	16,620	44	0.00	0.00
(32,12,2)	8,304	4,589	31	0.00	0.00
(32,24,2)	2,424,107	31,083	255	0.95	0.10
(8,12,3)	153	348	46	0.00	0.00
(8,24,3)	807	6,367	154	0.00	0.00
(16,12,3)	1,130	3,362	114	0.00	0.47
(16,24,3)	12,857	78,599	470	0.00	0.45
(32,12,3)	8,304	27,169	73	0.00	0.24

The perfect-information bound is precomputed once per bank. The continuous-count comparator uses warm or cold SLSQP calls and the global tangent certificate in Appendix F. A feasible primal point alone is not a certified upper success bound. The comparison checks convergence, constraints, tangent residuals, and iteration counts, and measures all time components. A high-precision check of optimizer witnesses provides an additional numerical check, not proof that floating-point code is formally sound. Greedy's actual executable value is computed separately; borrowing a library incumbent without charging for its construction is prohibited in the end-to-end runtime comparison.

H.3 Measured 10,000-query sequences

The timing study fixes the main moderate bank and draws 10,000 priors with seed `0x51555259`; the first 32 equal the representation-comparison priors. It executes that full sequence three times, rebuilding each library on each repeat. The convex solver warm-starts only from its own preceding solution. Grid lower/upper lookup, α lower/upper lookup, convex solve, and the common greedy incumbent are timed separately; each cumulative end-to-end method receives its own compilation and query cost plus the common greedy charge. No curve is extended outside the measured query range.

Figure 1 shows the complete measured region in the main text.

Across all 10,000 priors, maximum gaps are approximately 0.10 points for α pruning, 0.95 for the grid, and 0.36 for convex plus greedy. These differ from the 32-prior maxima because the query set is larger. The 32-prior grid value rounds to 0.95 points. Reported measurements use two decimal places. Compilation is amortized only over measured queries here. Every individual query and endpoint is measured, allowing alternative tolerance or cost accounting without extrapolating from 32 observations.

H.4 Support size and resource failures

For each $K \in \{2, 3, 4, 6, 8, 12\}$, the support study uses the first K coordinates of the same 16-action, 12-state parent bank. Costs are identical. There are 650 feasible count vectors at $B = 12$ and 42,083

at $B = 24$. Priors are 32 declared Dirichlet draws per support. The grid tries $Q = 16, 32, 64, 128$ and stops when the running lower maximum/upper minimum has width at most .01 on all these priors. This is not an empirical covering argument for the entire simplex. A target not reached within time/memory limits is recorded as failed, even when an earlier coarse library was completed.

The α comparison uses per-backup $\rho = .01/\lfloor B/c_* \rfloor$, giving a full-simplex theoretical error allowance, and reports its computed widths on the same priors. The table distinguishes the last completed grid stage from a successful final tolerance. The a priori record bound need not describe the actual number of stored records. Completed stages can vary between repeated evaluations under the same time cap: one evaluation completed the $K = 4, B = 24, Q = 64$ stage as well, but still missed the target. Common completed stages matched numerically; timing variation did not change the failure to meet the target.

Table 6: Every declared support/budget case, one percentage-point target. Gaps are points on 32 priors. Grid time is cumulative across completed refinements; † marks a completed coarse stage whose subsequent refinement hit a limit, so it did *not* meet the target. Unmarked finite entries meet it. α has a 20-second cap. Budget 12 has only 650 feasible counts, so successful $K = 12$ cases do not demonstrate large-bank scalability.

K, B	Grid records / time	Grid gap	α records / time	α gap
2,12	274 / 0.01s	0.00	54 / 0.08s	0.00
2,24	5,775 / 0.48s	0.59	213 / 0.38s	0.06
3,12	688 / 0.02s	0.19	103 / 0.13s	0.08
3,24	264,181 / 16.40s	0.89	751 / 1.50s	0.34
4,12	1,005 / 0.03s	0.00	101 / 0.16s	0.26
4,24	65,475 / 3.81s†	2.56	1,062 / 2.05s	0.25
6,12	740 / 0.01s	0.03	165 / 0.33s	0.43
6,24	470,953 / 17.22s†	1.38	1,843 / 7.14s	0.57
8,12	784 / 0.01s	0.00	248 / 0.41s	0.32
8,24	139,589 / 4.12s†	1.19	20s limit	–
12,12	1,539 / 0.03s	0.00	432 / 0.86s	0.34
12,24	Memory limit	–	20s limit	–

The grid can store more records than feasible count vectors: different action orders can round to different keys while yielding the same counts. This is not counted as a beneficial compression. α candidates are deduplicated by count witness. The empirical conclusion is not that either method handles arbitrary latent support. At the more demanding $B = 24$, ordinary α pruning reaches $K = 6$ here; both the current implementation and the theory retain real limits.

H.5 Two-bank phase compilation

The phase workload uses two-state banks, costs in $\{2, \ldots, 7\}$, interior completion vectors, a binary diagnostic of cost two, and a cost-three stochastic refresh. The seed is $178000 + 100m + B$, a fixed function of action count and budget. Seven cases use $m = 4, 8, 16, 32$ with $B = 12$, plus $m = 8, 16, 32$ with $B = 24$. Both lower and upper α planners are constructed and timed, so reported preparation includes duplicate construction with no assumed reuse between those constructions. Refresh kernels and diagnostics are known. Executable policies are evaluated independently in the original model.

The largest case has 16,855,983 feasible allocations *per bank*, 685 retained vectors across the two banks, and 77,498 total block evaluations across lower and upper recursion. Its bracket, rounded outward, is $[95.06\%, 95.38\%]$. No exhaustive runtime is asserted for that case. Separate random validation uses 40 generated two-bank models, two local pruning tolerances each, with a primitive action-level reference. Every bracket contains the reference optimum and each lower policy reproduces its claimed value. This tests phase composition of a conventional α oracle, rather than establishing that the vector representation itself is novel.

H.6 Random learned-checkpoint catalogs: generation and information

Each catalog has three width-three, six-internal-layer graphs with edge density selected from $.6, .7, .8, .9$. Calibration knows the layout label; deployment has a uniform prior over those three layouts and sees only paid diagnostic and refresh outcomes. Initial paths are the root and three prefixes of a sampled trajectory that fails in every layout. A finite declared refresh service samples a partial path and retains it and its parent, allowing banks with up to eight completion actions. Previously retained paths remain available. Checked refresh reports validity and pays for that check. Unchecked refresh creates the path without exposing validity, has lower specified cost, and permits later paid diagnostics.

Seeds 79000–79063 yield 64 catalogs, with 16 at each fixed density; all generated catalogs are included. Action-level checks use seeds 79000–79007; the other 56 catalogs use seeds 79008–79063. Both subsets use the same generator and six budget/quota settings. The seed law is not tuned using outcomes. Namespaced, range-checked Philox keys encode catalog, local checkpoint, and layout separately. Within one identifier, the first 32 samples are nested inside 128 and all policies use the same samples; this sharing is intentional. The 354,048 maximal-budget completions are counted once, not once per method, interface, or budget.

Models use $(S + .5)/(R + 1)$ with $R = 128$; neither those point estimates nor the learned graph study carry an acquired TV guarantee. Known diagnostic/refresh kernels and supplied three-layout support are part of the information interface. Exact integration computes true deployment success and expected work on that catalog without giving its hidden layout or true completion rates to the controller. Therefore this study assesses policy sensitivity to estimated rates within a declared catalog, not task-distribution transfer.

Baselines and the information they receive. Frozen forbids refresh, but may diagnose and use a mixed allocation. Global permits refresh but forbids further diagnostics *after the first refresh*. Greedy chooses a next primitive action using a residual-filling tail; One-step uses an optimized-count tail with the same available diagnostic and refresh lookahead. Both replan after observations. Phase solves the full finite-model phase recursion. Every comparison charges the same action costs and obeys the same (B, k, r) quotas. A baseline may decline a diagnostic; its name describes its available actions, not a compulsory purchase. All five use the same fitted rates and supplied kernels.

Uncertainty and avoiding pseudo-replication. The independent units are catalogs, not layouts, graph edges, policy variants, or reused budgets. For a paired difference d_{hi} in density stratum h, the estimate is $\frac{1}{4}\sum_h \bar{d}_h$, with standard error $\frac{1}{4}\sqrt{\sum_h s_h^2/n_h}$. The full sample has $n_h = 16$; the 56-catalog subset outside the action checks has $n_h = 14$. Stratified percentile bootstrap intervals use 10,000 resamples within each stratum and fixed reported seeds. They are descriptive, do not adjust for all displayed comparisons, and are not the finite-sample confidence guarantee of Theorem 6. All six settings are reported. The eight-catalog action-check subset, the remaining 56 catalogs, and all 64 catalogs are analyzed separately. These overlapping analyses reuse data and are not independent replications.

Diagnosis gains outside the action-check subset. The 56 unchecked catalogs outside the action-check subset show post-refresh diagnosis gains in the same unfiltered family, rather than only in the deliberate separation. At $(18, 2, 2)$, these 56 catalogs give a Phase-minus-Global mean of 0.59 points and a descriptive bootstrap interval of $[0.33, 0.87]$ points. At $(18, 1, 1)$ the mean is 0.44 points, and at $(30, 2, 1)$ it is 0.54. Checked differences remain very small. These effects are much less than the possibility construction's half-probability separation. The claim is evidence of a modest regime-dependent effect, not a universal need for post-refresh diagnosis.

H.7 Action-level diagnosis of tied baselines

A paired action check replays the fitted policies in the eight-catalog subset under common true outcomes until their first different primitive action. It integrates the probability of reaching such a disagreement, rather than inferring policy identity from rounded success. Exact reachable-policy identity occurs in $6, 6, 6$ of eight checked catalogs and $4, 7, 4$ of eight unchecked catalogs, for $(18, 1, 1), (18, 2, 2), (30, 2, 1)$ respectively. Mean first-disagreement probabilities are

$2.55\%, 2.31\%, 1.13\%$ and $11.11\%, 1.85\%, 19.95\%$. Tied aggregate values thus often reflect identical decisions, but not a universally identical implementation. We describe these as two tail variants of the same one-step family, not as independent evidence for two unrelated baselines.

Frozen purchases exactly one diagnostic with aggregate probability $7/8$ at budget 18 and $6/8$ at budget 30 in the eight-catalog action-check subset; otherwise it buys none. The second-diagnostic purchase probability is zero. The extra quota consequently does not improve those budget-18 policies, explaining the repeated 44.75% value. The full 64-catalog analysis also measures expected diagnostic and refresh counts for every method; the identity in the eight-catalog subset is not extrapolated to every catalog.

H.8 The non-monotone grid witness

For the moderate bank $(K, m, B) = (2, 8, 24)$ at seed 910824, costs are $(7, 5, 7, 2, 6, 4, 7, 6)$. Exhaustive enumeration gives 807 feasible count vectors. Across all 32 checked priors, the $Q = 32$ and $Q = 64$ libraries select identical executable counts and identical lower values, exact within 1.67×10^{-16}. Only the upper endpoint changes. At query 22, the endpoints are approximately

$$L \approx 36.54\%, \qquad U_{64} \approx 37.52\%.$$

The executable counts are $(0, 0, 0, 6, 0, 3, 0, 0)$ at cost 24. The optimistic key is $(44, 45)$ at the same cost. Its first coordinate comes from counts $(0, 0, 0, 4, 0, 4, 0, 0)$ with failure vector (shown to two decimal places) $(0.61, 0.67)$; its second comes from $(0, 0, 2, 1, 0, 2, 0, 0)$ with failure vector $(0.66, 0.65)$. Combining minima gives

$$(0.61, 0.65),$$

whose unrounded value no single same-cost plan realizes. An independent ordered-path computation reconstructs both coordinate witnesses without reading the production minima. The gap of 0.98 points is entirely optimism introduced by this nonnested path partition. It disappears at $Q = 128$. Retaining the running lower maximum and upper minimum yields monotone stopping certificates without asserting monotonicity of individual grids.

H.9 Sampled decoder: acquisition, not declared model perturbation

The closed target family contains four modes: little/big endian, signed/unsigned. Candidate code additionally chooses offsets from $\{-1, 0, 1, 256\}$, giving 16 implementations. The checker compares actual integer outputs with the mode's reference implementation on every byte pair. Seven declared proposal menus assign distributions to those implementations; the controller's calibration interface returns a sampled program and its checked accept/reject result, not the menu's exact success rate. The evaluator separately knows menu weights to integrate ground truth. This distinction emulates a black-box proposal distribution without pretending that the 16-candidate grammar itself is computationally inaccessible.

Four observed banks share these seven menus; new fragments are created by known refresh kernels. Calibration is labeled by mode. This is a stronger access assumption than unlabeled deployment, made explicit rather than hidden in a learned-state claim. Known deterministic probes identify aspects of the specification. The prior is $(.2, .3, .3, .2)$, budget 12, diagnostic quota two, refresh quota one. The four-state support is sufficient under the closed specification and proposal laws. It would not remain sufficient if independent per-menu persistent faults were introduced.

We collect 65,536 independent candidate proposals per menu/mode pair, recording which candidate is proposed and whether it is accepted. The keys use a distinct `0x56364443` namespace. Five listed prefixes, $R = 256, 1024, 4096, 16384, 65536$, share samples intentionally. All 140 row/size pairs receive simultaneous coverage through a single $\delta = .05$ family. True probabilities are not used to form the intervals, select the policy, or compute the bracket. They are used to check coverage in this one run and to evaluate the selected policy. Observed coverage in one run is not the proof of the statistical guarantee.

Table 4 in the main text reports every sample size.

At the maximum sample size, the largest acquired row error radius is 6.98×10^{-3} and its budget coupling allowance is 4.12×10^{-2}. The nominal fitted-model bracket, although nearly exact for its

supplied rates, must be enlarged to $[57.22\%, 65.46\%]$ by this generic route. Direct completion-rate monotonicity tightens it to $[60.71\%, 61.97\%]$. Thus model uncertainty, not rounding, is the main source of error in this application. The sample size is large for this small example. Exact semantic analysis could be cheaper here; the purpose is to demonstrate a complete statistical acquisition-to-guarantee loop with transparent access assumptions.

H.10 Fixed-bank allocation and expected-cost controls

The fixed-bank control compares residual filling with exact mixed allocation under matched diagnostic access. It uses three families: unplanted 3-SAT with 10 or 12 variables and width-three layered graphs with six internal layers. SAT formulas have a uniformly chosen clause count in $\{2v, 3v, 4v, 5v\}$; each clause samples three distinct variables and independent literal signs. Graph edges between consecutive layers are independent, with density chosen uniformly from $\{.45, .60, .75, .90\}$. Unsolvable tasks are retained. Each task starts with a uniformly sampled failed assignment or path; graph instances with no failed path are regenerated. Completions retain an anchor prefix and sample the suffix uniformly. SAT diagnostics apply unit propagation; graph diagnostics check retained edges and reachability through the next two edges. Both can detect impossibility but do not certify that a prefix is extendible.

The retained depths are $(0, 2, 6)$ for SAT-10, $(0, 3, 8)$ for SAT-12, and $(0, 2, 4)$ for graphs. Generation plus a terminal check gives costs $(11, 9, 5)$, $(13, 10, 5)$, and $(7, 5, 3)$, respectively. Budget triples are $(15, 30, 60)$, $(18, 36, 72)$, and $(14, 28, 42)$; diagnostic cost is two or five, with at most one diagnostic per policy. These choices give 18 conditions. Each family has eight independent pools, each with 300 training and 300 test tasks: 7,200 of each overall. For zero-based family, pool, and within-split task indices f, p, i, the seed is $9{,}000{,}000 + 2{,}000{,}000f + 10{,}000p + i$, plus 5,000 for test tasks.

Both methods use exact training-task rates and the same diagnostic outcomes to fit their plans; this is a supplied-rate control, not finite-row model acquisition. Exact mixed allocation enumerates feasible counts. Residual filling chooses the best training value among greedy marginal-gain and gain-per-cost allocations, including gain-per-cost filling after each affordable pure-action allocation. Both may decline a diagnostic and optimize outcome-specific plans. Test values integrate conditional success exactly, then average within each pool and across eight pools. Budgets and diagnostic prices reuse the same tasks, so the 18 conditions are not independent datasets. Residual filling ties exact mixed allocation in 14/18 conditions, including every graph condition; the four SAT differences are only 0.10–0.27 points. Absolute SAT success is commonly 2–17%. These negative controls limit claims about the practical need for exact mixed allocation.

The same 18 conditions and eight pools give 144 fitted diagnostic-dependent count maps for the expected-cost control. A compiler reorders a fixed map, potentially deferring or omitting its diagnostic, while preserving deadline success. Of the 144 fits, 134 buy a diagnostic; no compiled execution defers or omits it, and only within-leaf ordering saves 2.25×10^{-3} abstract units on average. This comparison concerns execution of a fixed count map, not optimization over all policies. It illustrates why preserving deadline success need not imply an improvement in expected expenditure.