RISK-CONTROLLED SELECTIVE LLM ANSWERING BY PRICING LABEL-FREE CHECKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Serving an answer from a large language model requires deciding when to abstain, yet a verifier's ranking accuracy alone does not determine the error rate among served answers. We introduce PriceCheck, which builds a compact family of decision rules from label-free checks such as re-solving a problem. Each check has a price: its agreement rates on correct and incorrect answers and its cost per run. Prices fitted on a small, class-enriched labelled set compose into predictions of a schedule's coverage and cost, guiding which checks to run and when to stop. A calibration test then selects a schedule at a stated selective-risk target. In mathematics, the selected schedules serve 76.1% of answers on average and keep held-out selective risk below 1.5% on all 15 splits. Under the shared testing protocol, PriceCheck serves more answers at that target than reward models, a prompted judge, the generator's confidence and a trained correctness classifier. At matched coverage, it keeps the fewest wrong answers among these scorers. Across 118 diagnostic schedules, price-based coverage predictions have a rank correlation of 0.97 with observed coverage. These results show that choosing how checks are combined and stopped matters alongside how well a verifier ranks answers.

1 INTRODUCTION

Large language models increasingly rely on a verifier to choose among the candidate answers they generate. The verifier may take a majority vote among the candidates (Wang et al., 2022), apply a trained reward model (Cobbe et al., 2021; Uesato et al., 2022; Lightman et al., 2023; Wang et al., 2024; Zhang et al., 2024), or prompt a model to judge the work (Zheng et al., 2023; Weng et al., 2023). The resulting ranking is enough to pick a winner. However, it does not tell a deployed system how often the answers it serves are wrong, a quantity known as *selective risk*. Selective prediction studies when to abstain so as to control this risk (Geifman & El-Yaniv, 2017; Kamath et al., 2020), and distribution-free risk control can certify that it stays below a stated target on held-out calibration data (Bates et al., 2021; Angelopoulos et al., 2022; 2021). Evaluations of verifier rankings (Zhang et al., 2025) therefore leave a further design question: which checks should a system run, combine and stop to serve many answers at a stated risk target?

Connecting ranking to selective answering requires choosing the rules to test. In a finite-family application of Learn-then-Test, candidate rules are fixed independently of calibration data, and the multiplicity correction grows with the number of rules (Angelopoulos et al., 2021). The family should therefore be small and well chosen. Available label-free checks differ both in how often they agree with wrong answers and in what they cost. A ranking alone does not specify which checks to combine, in what order, or when to stop.

Figure 1 summarises PriceCheck, our approach to pricing and scheduling these checks. A *check* reads a candidate answer, but not its label, and returns a binary verdict on whether it agrees with that answer. Examples include a re-solve that reaches the same answer (Wang et al., 2022) and a backward probe that recovers an answer from reconstructed reasoning (Weng et al., 2023). We attach a *price* to each check: its *completeness*, or agreement rate on correct answers; its *leak*, or agreement rate on wrong answers; and its *unit cost*. We fit the rates on a labelled pricing set.

A *schedule* uses one or more checks to serve an answer or abstain; its *coverage* is the share of answers served. Composing check prices gives estimates of schedule coverage and cost. This includes cascades that pass a cheap check's undecided cases to a costlier check. We test a declared family

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

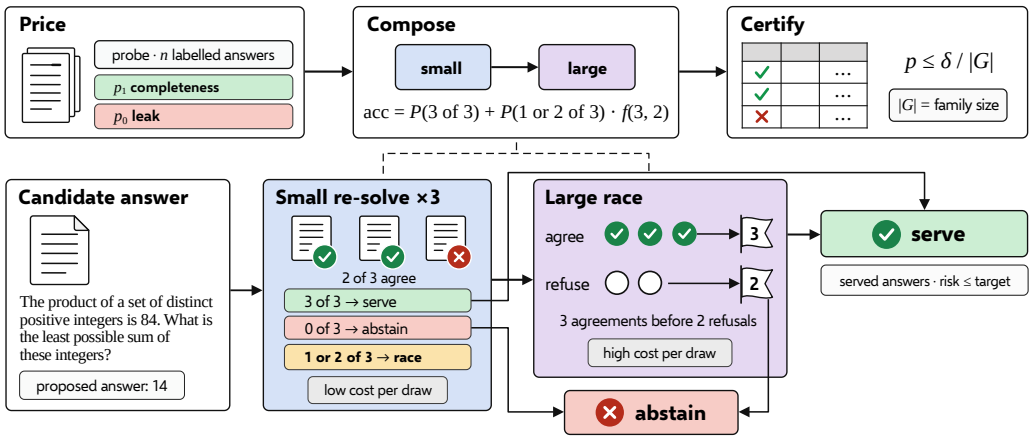

Figure 1: Overview of PriceCheck. Top, the three steps: price each check on a labelled probe, compose prices into a schedule, and certify one schedule. Bottom, one candidate answer passing through a two-stage cascade. Green serves, red abstains, and amber passes the undecided band on.

on calibration data and select a schedule at a stated selective-risk target (Angelopoulos et al., 2021). Pricing thus connects measurements of individual checks to the choice of a complete answering rule.

We evaluate PriceCheck on answers to competition mathematics problems (Hendrycks et al., 2021) produced by five fine-tuned generators. Under the shared calibration-testing protocol, we compare with process and outcome reward models, a prompted judge, the generator's confidence (Kadavath et al., 2022), and a classifier trained on the calibration labels. Across 118 diagnostic schedules, price-based coverage predictions have a rank correlation of 0.97 with observed coverage. At the primary 1.5% target, the selected schedules serve 76.1% of answers on average, keep held-out risk below the target on all 15 splits, and retain fewer wrong answers than every rival scorer at matched coverage.

We also examine the dependence among answers to the same problem (Kish, 1965). Counting each served problem once widens the nominal risk bounds while preserving the advantage of the selected re-solve votes over the trained process reward model. Together, these findings connect the choice of check schedules to coverage and observed risk, beyond the ranking quality of a verifier alone.

2 Related Work

Verifiers that order candidates. Step-level process reward models learn from step annotations and return a score per solution (Uesato et al., 2022; Lightman et al., 2023; Wang et al., 2024; Zhang et al., 2025; Luo et al., 2024; Chen et al., 2027), whereas outcome-level models score the finished answer, some of them writing the verdict as text (Cobbe et al., 2021; Dong et al., 2024; Zhang et al., 2024). Prompting a capable model to grade another's work needs no annotation at all (Zheng et al., 2023; Yamauchi et al., 2026), and the generator's own confidence in what it wrote is cheaper still (Kadavath et al., 2022; Tian et al., 2023; Kuhn et al., 2023; Manakul et al., 2023; Panov et al., 2026), although models do not reliably notice their own errors (Huang et al., 2023; Stechly et al., 2023; Lee & Park, 2026). An ordering selects a winner but does not name an operating point, and, as our experiments show, it is also a poor guide to what a scorer can certify. We therefore learn no score and instead measure two rates per check, fitted once on a small labelled probe.

Checks and sampling controllers. Our backward probe descends from checks that re-derive something the answer ought to imply, such as a masked condition (Weng et al., 2023), a masked quantity (Jiang et al., 2024), the premises a chain of thought assumed (Xue et al., 2023), or the specification a program was written from (Allamanis et al., 2024). Related checks read the derivation as written (Ling et al., 2023; Madaan et al., 2023). In parallel, sampling controllers spend a budget rather than pass a verdict, through a fixed vote over sampled solutions (Wang et al., 2022), variants that stop once the vote settles (Aggarwal et al., 2023; Li et al., 2024), and studies of how to allocate the budget (Brown et al., 2024; Snell et al., 2024; Zuo et al., 2026; Lee et al., 2026a). However, both

return a decision without the rates that would predict how a chain of checks behaves or which check is worth its cost. We split agreement into its rate on correct and on wrong answers, charge a unit cost on three axes, and predict the coverage of a schedule, cascades included, before it runs.

Distribution-free guarantees. Serving an answer only when a score clears a threshold is selective prediction (Chow, 1970; El-Yaniv & Wiener, 2010; Geifman & El-Yaniv, 2017; Kamath et al., 2020), and distribution-free calibration turns such a threshold into a finite-sample statement about a predictive set (Vovk et al., 2005; Angelopoulos & Bates, 2021), a risk functional (Bates et al., 2021; Angelopoulos et al., 2022), or a family of configurations tested at once (Angelopoulos et al., 2021). Applications abstain on a generation or filter the claims inside one (Quach et al., 2023; Yadkori et al., 2024; Nguyen et al., 2026). We build on this machinery rather than replace it. The binomial test assumes independent calibration answers and takes the candidate family as given. PriceCheck uses prices to construct that family and assesses within-problem dependence with problem-count risk bounds.

3 PRICECHECK

We first define the price of a check, then how prices compose into schedules, and finally the certificate. Throughout, a *candidate answer* is one stored pair of a problem and a proposed answer, and an *action*, our term for a check, returns a single binary observation on a candidate answer, which records whether it agrees with the proposal. No action reads a correctness label when it runs.

Problem formulation. Let $x \in \mathcal{X}$ denote a problem, $\hat{y} \in \mathcal{Y}$ the proposed answer a generator produces for it, and $y^* \in \mathcal{Y}$ the ground-truth answer, so that $(x, \hat{y})$ is a candidate answer. We write its latent correctness indicator as $Y = \mathbb{I}(\hat{y} = y^*)$. A *selective rule* $S(x, \hat{y}) \in \{0, 1\}$ serves the candidate answer when $S = 1$ and abstains when $S = 0$. Schedules in Section 3.2 build such rules from actions. A rule is evaluated by its coverage $C(S) = \mathbb{P}(S = 1)$ and its selective risk

$$R(S) \;=\; \mathbb{P}(Y = 0 \mid S = 1), \tag{1}$$

the error rate among served answers. Given a selective-risk target $\alpha \in (0, 1)$ and a confidence budget $\delta \in (0, 1)$, our objective is to select and certify a schedule S for which $\mathbb{P}\big(R(S) \le \alpha\big) \ge 1-\delta$, where the probability is over the draw of calibration data, at the largest coverage $C(S)$ we can obtain.

3.1 THE PRICE OF AN ACTION

We define the *price* of an action as two measured rates and a unit cost. Writing $V \in \{0, 1\}$ for the verdict of one draw, the *completeness* $p_1 = \mathbb{P}(V = 1 \mid Y = 1)$ is the per-draw rate at which the action agrees with a correct answer, and the *leak* $p_0 = \mathbb{P}(V = 1 \mid Y = 0)$ is the per-draw rate at which it agrees with a wrong one. The *unit cost* is the cost of one draw. Table 1 lists both rates for our seven actions, together with the unit cost on two of our three cost axes, generated tokens and parameter-weighted tokens. The two axes do not order the actions identically, so we carry both through the analysis.

Backward probe. Given the problem and the proposed answer, a reverse model writes the reasoning that leads to that answer, and a forward model then recovers an answer from that trace (Weng et al., 2023; Lee et al., 2026b). Each draw is one such trace and recovery, and its verdict is whether the recovered answer agrees with the proposed one. We take $K = 3$ draws and record the agreement count $m \in \{0, \ldots, K\}$. We run the probe at three model sizes, 8B, 4B and 1.7B, and give both prompts verbatim in Appendix A.3.

Re-solve vote. For every distinct problem, we draw fresh solutions from the frozen forward model, 16 at 8B and 8 at 1.7B, and each draw casts a vote whose verdict is whether its answer agrees with the proposed answer (Wang et al., 2022). A draw with no extractable answer counts as disagreement, since a schedule must not serve an answer on a verdict it cannot read.

Generator confidence. Two further actions read the generator's own confidence: its probability of Yes when asked whether the proposed answer is correct, and the mean log-probability of the stored solution (Kadavath et al., 2022). We set each threshold to maximise completeness minus leak on the full pool, defining a descriptive one-draw action charged one scoring pass with no generated tokens.

Table 1: Price of each action on the frozen pool: completeness p_1 and leak p_0 per draw, and unit cost per draw in generated and in parameter-weighted tokens. The two confidence actions re-score the stored solution, so they generate nothing.

	Backward probe			Re-solve vote		Generator confidence	
	8B	4B	1.7B	8B	1.7B	p(True)	Log-prob.
Rate per draw (%)							
Completeness p_1	70.1	73.1	63.9	80.2	79.5	72.5	61.9
Leak p_0	10.8	13.8	12.5	9.0	10.6	54.5	30.3
Cost per draw (tokens)							
Generated	1034	1034	1034	5384	4801	0	0
Weighted	1034	517	217	5384	1008	0	0

Fitting the price. Our prediction experiments fit p_1 and p_0 on a class-enriched pricing set of 100 labelled candidate answers, stratified by generator with at least eight wrong answers. Appendix A.10 gives the sampling recipe and overdispersion estimate; Table 1 reports descriptive rates on the full pool.

3.2 Schedules over priced actions

A *schedule* maps the verdicts of one or more actions to serve or abstain. We build schedules from four primitives. A *threshold* serves when the agreement count m of one backward probe reaches t. *Unanimity* serves when the first N re-solve votes all agree. An *adaptive race* draws re-solve votes until it reaches k agreements, where it serves, or j disagreements, where it abstains. A *cascade* runs a cheap action first, serves or abstains on its clear outcomes, and sends the remaining undecided band to the race, as in the lower half of Figure 1. Our deployed family G is a fixed grid of 23 schedules that no certification run alters, and the nested protocol of Section 4.3 charges for how we chose its members.

Composing prices. Pricing actions separately on correct and wrong answers lets us predict a schedule's coverage, selective risk and cost. Let a_y be its fitted acceptance probability for class y, where $y = 1$ marks a correct answer and $y = 0$ a wrong one. We use the correct-answer fraction π of the class-enriched pricing set as a plug-in prior, giving

$$\hat{c} \;=\; \pi\, a_1 + (1-\pi)\, a_0, \qquad \hat{r} \;=\; \frac{(1-\pi)\, a_0}{\hat{c}}. \tag{2}$$

For a threshold t on an action with K draws, a_y is the binomial tail $\mathbb{P}(m \geq t \mid K, p_y)$, replaced by a beta-binomial tail when the fitted counts are overdispersed. An adaptive race that serves at k agreements and abstains at j disagreements serves with probability $f_y(k, j)$, given by the recursion $f_y(k,j) = p_y\, f_y(k-1, j) + (1-p_y)\, f_y(k, j-1)$ with $f_y(0, j) = 1$ for $j > 0$ and $f_y(k, 0) = 0$ for $k > 0$. A cascade whose first stage serves when all K draws agree, abstains when none does, and passes every other count to the race, therefore, serves with

$$a_y \;=\; \mathbb{P}(m = K \mid K, p_y) \;+\; \mathbb{P}(0 < m < K \mid K, p_y)\; f_y^{\mathrm{v}}(k, j), \tag{3}$$

where p_y are the rates of the first-stage action and f_y^{v} is the race recursion evaluated at the rates of the re-solve vote. Equation (3) treats the two stages as independent given the label, an assumption whose cost we measure in the analysis.

Cost composes in the same way: each stage is paid only by the answers that reach it, so a first-stage abstention pays for that stage alone. Appendix E.5 evaluates the resulting cost predictions on both token axes.

3.3 Certification

With i.i.d. calibration answers from the deployment population and a family G fixed independently of them, each schedule $S \in G$ yields an exact one-sided binomial p-value against $H_0 \colon R(S) > \alpha$ (Angelopoulos et al., 2021),

$$p \;=\; F(e;\, n,\, \alpha), \qquad \text{certified} \iff n > 0 \;\text{ and }\; p \leq \delta/|G|, \tag{4}$$

where n is the number of calibration answers S serves, e is how many of those are wrong, F is the binomial cumulative distribution function, $\delta = 0.05$ is the confidence budget and $|G| = 23$ is the declared family size, so that $\delta/|G|$ is a Bonferroni correction. Our clustered and score-adaptive comparisons apply this test at its nominal level and assess realised held-out risk; Section 5 examines dependence. Among schedules passing the test, a maximum-coverage selector takes the one with the largest calibration coverage, and a cost-minimising selector takes the cheapest one whose calibration coverage is at least 0.60. The maximum-coverage selector is the default, and we report the selected schedule's coverage, selective risk and cost on the test side, which certification never reads. Our primary target is $\alpha = 0.015$, the strictest level at which half of the 20 repeats of the nested protocol in Section 4.3 certify, against a quarter at 1% and none at 0.5%.

For a fair comparison, each rival scorer receives a family of selective rules of the same shape and comparable size. We aggregate its step scores by product, minimum, mean or last step, and cross each aggregation with six abstention levels set at quantiles of the calibration scores, never of the labels. We also certify, for each rival, a family that fixes one aggregation and sweeps 24 levels, marked with a dagger (†) in Table 2; where the aggregation matters, we fix the one that certifies the most coverage, which favours the rival.

The divisor $|G|$ counts the distinct rules a family contains before any label is read (Dunn, 1961). For a step-level scorer, every pair of aggregation and level is a distinct rule, which gives 24. In contrast, a scorer that returns one scalar per answer makes its four aggregation columns identical by construction, so its family is the six levels alone, and charging 24 would over-correct it fourfold.

Finally, when a population share π_{pop} of candidate answers is correct, no selective rule can serve more than $\min\{1, \pi_{\text{pop}}/(1-\alpha)\}$ at target α (Chow, 1970), which caps coverage at 97.20% on our pool at the primary target.

4 Main Results

Table 2 compares PriceCheck with a prompted judge (Yang et al., 2025), a trained process reward model (PRM) (Yang et al., 2024), Math-Shepherd (Wang et al., 2024), and an outcome reward model (ORM) (Xiong et al., 2024). We also evaluate the generator's own confidence. All methods use the shared answer-level testing protocol of Section 3.3 across fifteen splits, with the same selective-risk targets and confidence budget.

Setting. Five generators, F0, STaR, SelfDistill, ForwardPref and iter2, all fine-tuned from one 8B backbone, each attempted the 500 problems of MATH-500 (Hendrycks et al., 2021). The 2331 attempts with an extractable answer form our frozen pool, 99 of them wrong. These candidate answers remain fixed throughout evaluation. Excluding five rows without a complete vote bank leaves 2326 answers, including all 99 wrong answers, with a base error of 4.26%. Appendix A.1 accounts for every excluded attempt.

Coverage is the share of answers a rule serves, and selective risk is the share of served answers that are wrong. Cost averages all answers, including those rejected. We use five source-held-out folds, each reserving one generator, and ten question-disjoint halves that reserve problems. Appendix A.9 gives split sizes and certification constants.

4.1 Certification at the stated target

Table 2 shows that PriceCheck certifies on all fifteen splits at the primary 1.5% target, with the highest mean held-out coverage, 76.09%, and no test-risk exceedance. Its coverage advantage over the best rival grows from 4.18 points at the primary target to 36.91 points at 0.75%. At the strictest 0.5% target, only PriceCheck certifies any split. The judge's finer family leads at the looser 2% target; priced schedules retain more coverage when tighter targets demand stronger calibration evidence.

Table 2 also shows which families certify on all fifteen splits. Among these, PriceCheck has the least dispersed coverage: a standard deviation of 4.4 points against 5.3 to 10.1 for the four rival families. For other rows, non-certifying splits counted as zero dominate the spread. Appendix C gives results by target and split.

Table 2: Certification under a shared protocol. At each target α, splits certifying of 15, coverage over all 15 counting non-certifying splits as zero, and wrong answers kept at matched coverage. A dagger marks 24 levels, not six. Bold is best.

	$\alpha = 0.5\%$		0.75%		1.0%		1.25%		1.5%		2.0%		Matched
Method	Spl.	Cov.	Spl.	Cov.	Spl.	Cov.	Spl.	Cov.	Spl.	Cov.	Spl.	Cov.	Wr.↓
PriceCheck (ours)	**5**	**23.20**	**12**	**57.00**	**15**	**72.11**	**15**	**73.85**	**15**	**76.09**	**15**	78.86	**44**
Prompted verifier													
Judge	0	0	5	16.74	**15**	52.43	**15**	62.32	**15**	69.50	**15**	78.17	60
Judge†	0	0	5	20.09	**15**	61.08	**15**	67.99	**15**	71.91	**15**	**80.81**	
Trained verifiers													
PRM	0	0	1	3.31	5	16.71	8	26.77	**15**	55.90	**15**	68.14	97
PRM†	0	0	1	3.90	8	32.09	**15**	62.46	**15**	66.55	**15**	74.20	
Math-Shepherd	0	0	0	0	0	0	0	0	1	3.56	5	17.52	164
Math-Shepherd†	0	0	0	0	0	0	0	0	2	7.61	6	23.80	
ORM	0	0	0	0	0	0	1	3.47	3	10.23	11	38.63	195
ORM†	0	0	0	0	0	0	0	0	2	7.66	9	34.94	

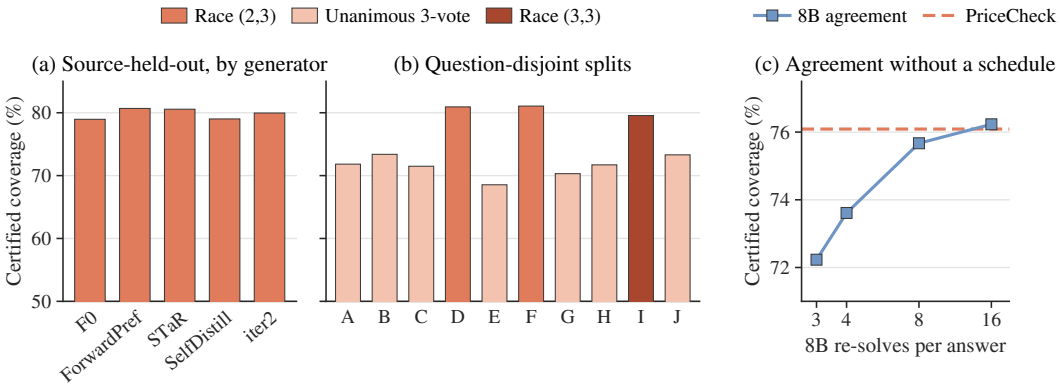

Figure 2: Held-out coverage certified by PriceCheck at the 1.5% target on (a) source-held-out folds and (b) question-disjoint halves A–J, coloured by the selected configuration. Race (k, j) serves at k agreements before j disagreements. (c) Coverage certified at 1.5% by 8B re-solve agreement over n re-solves per answer, against PriceCheck (dashed).

Ranking quality and certifiability. Table 6 shows that both generator-confidence scores rank answers above chance, yet neither certifies on any split at any target under either family. At matched coverage, the probability of Yes and mean log-probability keep 372 and 295 wrong answers, respectively, the most of any scorer (Kadavath et al., 2022). They are therefore omitted from Table 2. Above-chance ranking alone need not provide a cut point whose calibration evidence survives the family's multiplicity correction.

Across split schemes. Figure 2a,b shows higher coverage on source-held-out folds than on question-disjoint halves. Holding out problems costs the judge roughly twice as much coverage, which falls from 76.16% to 66.16%. The colours show that the maximum-coverage selector certifies an 8B re-solve vote on every split. The cost-minimising selector instead picks a small-first cascade on three of the five source-held-out folds, and its selections there average 77.7% coverage at under a third of the adaptive vote's weighted cost. Appendix C.2 lists the selection on every split.

Re-solve votes without a schedule. Figure 2c compares PriceCheck with the plain fraction of stored re-solves agreeing with an answer, certified as a scalar score under the same protocol. With three 8B re-solves, which generate about as many tokens as the certified schedule, this score certifies on fewer splits at the two strictest targets, serves less at the primary target and keeps more wrong answers at matched coverage. Stopping the race once its outcome is settled therefore certifies more from the same bank at the same generation budget, and the plain score reaches the schedule's coverage only at sixteen votes. Appendix C.3 reports every vote count at both model sizes.

Table 3: Design privilege at the 1.5% target. (a) Reward-model families of 24 rules, fixed in advance or distilled from a 508-rule survey. (b) A nested protocol charging both sides for design, a repeat certifying nothing counting as zero. Candidates gives the search-space size. QD: question-disjoint; SHO: source-held-out.

(a) Matched family size

		Coverage (%)		
Family	Rules	All	QD	SHO
PriceCheck	23	**76.09**	**74.21**	**79.84**
PRM, fixed	24	66.36	65.66	67.75
PRM, distilled	24	66.73	65.57	69.04

(b) Nested protocol, 20 repeats

			Coverage (%)	
Family	Candidates	Splits	Certifying	All
PriceCheck	87	**10/20**	**76.33**	**38.17**
PRM	508	8/20	64.18	25.67

4.2 Wrong answers kept

Matched coverage. Table 2 shows that every rival scorer keeps more wrong answers than PriceCheck at matched coverage over the fifteen splits. We rank each rival's test answers and keep exactly as many as the certified schedule kept on that split. Both families of a scorer share this count because it depends only on the ranking. Even the judge, the strongest ranker, keeps 60 wrong answers against PriceCheck's 44.

4.3 Design privilege

Matching design freedom. Table 3 shows that PriceCheck's coverage advantage persists when the reward model receives comparable family-design freedom. We first distil a 24-rule family for the process reward model from a 508-rule survey by the same kind of rule that built our grid. The reward model still certifies on every split, but the distilled family gains it under half a point on the fifteen-split mean, and it stays about nine points behind on question-disjoint halves and eleven on source-held-out folds. Both fixed families exceed the joint aggregation-by-six-level PRM family in Table 2, which serves 55.90%.

Table 3b tests family selection in a nested protocol. It re-selects each family on a design split of the problems and certifies it on a separate calibration split, which charges both sides for the design search. Under this protocol PriceCheck certifies on more repeats, and it serves more on each of the four repeats where both arms certify. Appendix C.4 gives the protocol, and Appendix B specifies each rival's construction.

A verifier trained on the certificate's own labels. Table 7 shows the comparison with a verifier fitted to the calibration labels. On each calibration side, we cross-fit a 1.7B correctness classifier in four folds, score the test side with the resulting ensemble, and certify it under the identical protocol (Yang et al., 2025). Its ranking improves steadily as the context window widens, yet even the widest window certifies on only three of the fifteen splits and keeps far more wrong answers at matched coverage, so widening what the scorer reads does not close the gap.

4.4 Controls on the backward probe

Figure 3a shows that re-pairing the saved recovered answers across problems reduces coverage to about one percent over a thousand trials for every generator (Ernst, 2004). Thus the stored agreements are specific to the candidate problem. A question-only re-solve with the same three-sample count and a 1536-token cap serves about a third as many answers (Wang et al., 2022); Appendix G.2 also reports a longer-cap control.

Figure 3b shows that removing either the proposed answer or the reverse trace reduces total acceptances by 40 to 53 percent on the 100-answer probe. None of the four ablated settings accepts a wrong answer. Both prompt components therefore contribute to agreement, and neither alone reproduces the full prompt's acceptance set. Appendix G reports each control in full.

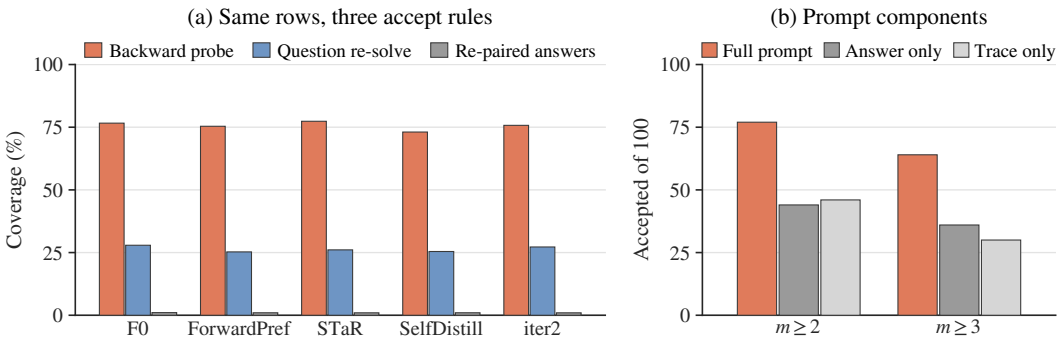

Figure 3: (a) Coverage on the same answers of the backward probe at one agreement, of a question-only re-solve with three samples capped at 1536 tokens, and of saved recovered answers re-paired at random across problems. (b) Acceptances on a 100-answer probe with one prompt component removed.

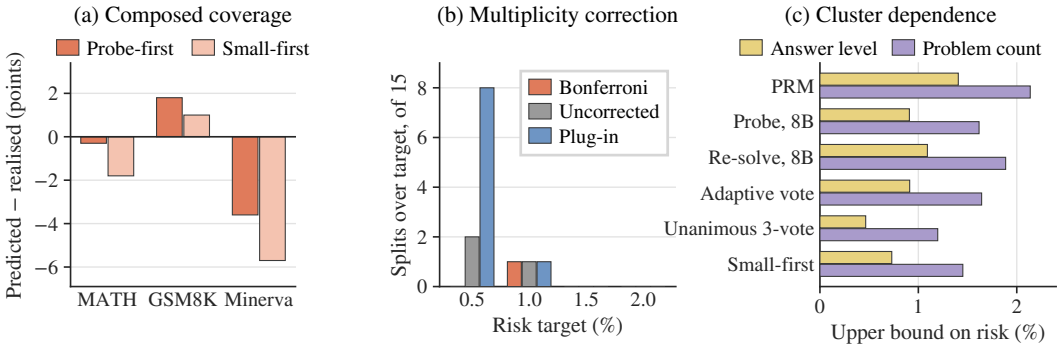

Figure 4: (a) Predicted minus realised coverage of two composed cascades on three answer pools, from a probe of 100 labels. (b) Splits, of 15, whose certified selection exceeds the target on test under three testing rules. (c) Nominal answer-level and problem-count 95% upper bounds on selective risk.

5 Analysis

Prices predict schedule coverage. Figure 4a shows that prices fitted on 100 class-enriched answers track cascade coverage across three pools. Table 16 reports a coverage rank correlation of 0.97 over 118 schedules and a mean absolute coverage error of 2.97 points over 70 cascades. Predictions are evaluated on the remaining answers in the same frozen pool. This diagnostic grid is separate from the 23-member certification family. Risk prediction has a mean error of about one point but is less reliable across schedules, so PriceCheck tests schedules on calibration data rather than using predicted risk as a certificate.

Table 13 shows that coverage error falls by 40 to 70 percent as the fitting set grows from 50 to 200 answers on each of five pools. For nine single-action thresholds, fitting on all 2326 answers reduces mean error to 0.25 points. Appendix E.5 distinguishes this sampling effect from the residual errors of composed cascades.

Table 16 also provides a reference for the independence approximation in Equation (3). Across 68 two-stage cascades, replacing the factorised composition with the exact one moves coverage by 0.92 points on average, about a third of the cascade prediction error, with a sign opposite to the prediction bias.

Prices across populations. Figure 5a,b shows how both rates shift across actions on two further mathematics benchmarks and on program synthesis (Cobbe et al., 2021; Lewkowycz et al., 2022; Chen et al., 2021). On every population, each action agrees with correct answers clearly more often than with wrong ones, so the checks remain informative even where the leak rises with the base error, and a price refitted on a new population is still a usable input to the same composition. On a pool from a different model family, the single-probe price also predicts realised coverage to within about four points (Grattafiori et al., 2024). Appendix E.3 lists every rate, and Appendix E.4 reports the transport.

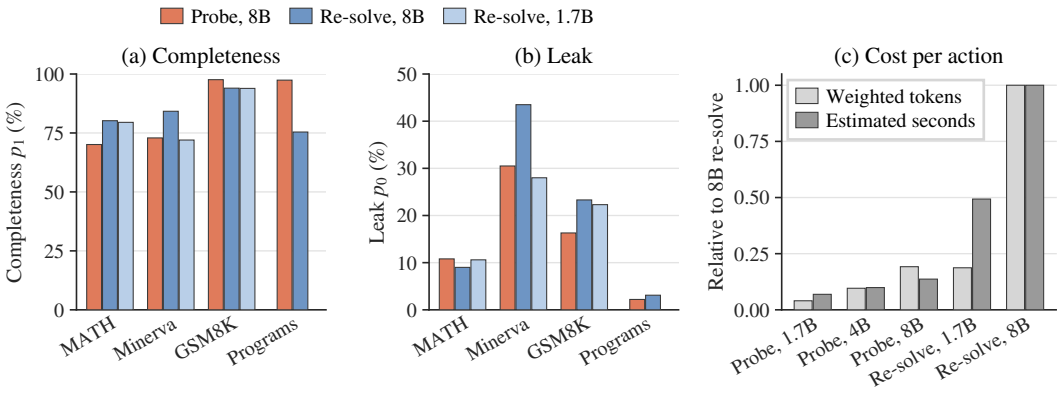

Figure 5: (a) Completeness and (b) leak per draw of three actions on four answer populations, where on programs the re-solve regenerates the program. (c) Parameter-weighted tokens and accelerator-time estimates from measured throughput, relative to an 8B re-solve.

The Bonferroni correction. Figure 4b shows that, at the strictest target, an uncorrected p-value and a plug-in rule that admits any schedule whose calibration risk is at most the target both certify every split but exceed the target on test, whereas the flat Bonferroni correction certifies a third of the splits and exceeds on none (Dunn, 1961). The flat divisor costs most when calibration keeps one answer per problem. In that regime, fixed-sequence testing, which tests rules one at a time at the full budget along an order fixed in advance and stops at the first failure, recovers part of the loss, and a price fitted on problems disjoint from calibration licenses cells at the primary target where flat Bonferroni licenses none. Appendix C.5 reports every order.

The three cost axes. Figure 5c shows that the parameter-weighted proxy overstates the small model's advantage and reverses the ordering of a 1.7B re-solve and an 8B probe. Table 18 estimates that the pooled small-first cascade uses about a third of the adaptive 8B vote's weighted tokens and about 30% less accelerator time. The time estimates scale measured throughput by stored token counts (Kwon et al., 2023). We report them alongside the parameter-weighted proxy commonly used for test-time compute (Snell et al., 2024); Appendix F gives all three axes.

Dependence between answers to one problem. Figure 4c compares nominal answer-level and problem-count risk bounds. The pool's error indicator has an intraclass correlation of 0.45 (Kish, 1965), and counting distinct kept problems widens every reported bound while preserving their ordering (Crowder, 1978; Rao & Scott, 1981). The selected re-solve votes retain a smaller upper bound than the trained reward model. In the beta-binomial simulations, from this correlation up to perfect within-problem dependence, the problem-count construction rejects a true null at most 3.6% of the time at a nominal 5% level (Field & Welsh, 2007). Appendix D reports every bound and the simulation.

6 Conclusion

We introduced PriceCheck, which constructs selective answering rules by pricing label-free checks and composing their agreement rates and costs. Prices fitted on 100 class-enriched labelled answers ranked the coverage of 118 diagnostic schedules with a correlation of 0.97. At the primary 1.5% target, the selected schedules served 76.1% of answers on average and kept held-out risk below the target on all 15 splits. Under the shared testing protocol, they served more answers than the rival scorers and retained the fewest wrong answers at matched coverage. These findings show that the design of check combinations and stopping rules matters alongside verifier ranking quality when deciding which answers to serve.

Reproducibility Statement

Appendix A specifies the answer pools, prompts, sampling settings, matching rules, cost accounting and calibration splits. Appendix B describes the trained and prompted scorers. Appendix C

gives the certification results, Appendix D specifies the dependence diagnostics and simulations, and Appendix E describes price estimation. The shared-pool comparisons replay stored check observations; additional-population evaluations and throughput measurements use the protocols stated in the appendix.

AI Use Statement

A large language model assisted with LaTeX tables, and language polishing. The authors reviewed and verified all outputs and take full responsibility for the paper.

Ethics Statement

This study evaluates answers to public mathematics and program-synthesis benchmarks using models based on publicly released weights. It involves no human participants or personal data. A deployment should communicate its observed error and abstention rates to the people who rely on its answers and handle unanswered questions.

References

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. Let's Sample Step by Step: Adaptive-Consistency for Efficient Reasoning and Coding with LLMs, 2023. URL http://arxiv.org/abs/2305.11860.

Miltiadis Allamanis, Sheena Panthaplackel, and Pengcheng Yin. Unsupervised Evaluation of Code LLMs with Round-Trip Correctness, 2024. URL http://arxiv.org/abs/2402.08699.

Anastasios N. Angelopoulos and Stephen Bates. A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification, 2021. URL http://arxiv.org/abs/2107.07511.

Anastasios N. Angelopoulos, Stephen Bates, Emmanuel J. Candès, Michael I. Jordan, and Lihua Lei. Learn then Test: Calibrating Predictive Algorithms to Achieve Risk Control, 2021. URL http://arxiv.org/abs/2110.01052.

Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal Risk Control, 2022. URL https://arxiv.org/abs/2208.02814.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Stephen Bates, Anastasios Angelopoulos, Lihua Lei, Jitendra Malik, and Michael Jordan. Distribution-free, Risk-controlling Prediction Sets, 2021. URL https://doi.org/10.1145/3478535.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large Language Monkeys: Scaling Inference Compute with Repeated Sampling, 2024. URL http://arxiv.org/abs/2407.21787.

Kaiyan Chen, Yimiao Qiu, Wang Lin, Qi Xia, and Jingyuan Chen. Reject-before-aggregate reward modeling for curriculum-constrained mathematical reasoning, 2027. URL https://doi.org/10.1016/j.neucom.2026.135077.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

C. Chow. On optimum recognition error and reject tradeoff, 1970. URL https://doi.org/10.1109/tit.1970.1054406.

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, 2021. URL http://arxiv.org/abs/2110.14168.

Martin J. Crowder. Beta-binomial ANOVA for proportions. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 27(1), 1978. doi: 10.2307/2346223.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. RLHF Workflow: From Reward Modeling to Online RLHF, 2024. URL https://arxiv.org/abs/2405.07863.

Olive Jean Dunn. Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293):52–64, 1961. doi: 10.1080/01621459.1961.10482090.

Ran El-Yaniv and Yair Wiener. On the Foundations of Noise-free Selective Classification, 2010. URL https://jmlr.org/papers/v11/el-yaniv10a.html.

Michael D. Ernst. Permutation methods: A basis for exact inference. *Statistical Science*, 19(4), 2004. doi: 10.1214/088342304000000396.

C. A. Field and A. H. Welsh. Bootstrapping clustered data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(3):369–390, 2007. doi: 10.1111/j.1467-9868.2007.00593.x.

Yonatan Geifman and Ran El-Yaniv. Selective Classification for Deep Neural Networks, 2017. URL http://arxiv.org/abs/1705.08500.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, et al. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset, 2021. URL https://arxiv.org/abs/2103.03874.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large Language Models Cannot Self-Correct Reasoning Yet, 2023. URL http://arxiv.org/abs/2310.01798.

Weisen Jiang, Han Shi, Longhui Yu, Zhengying Liu, Yu Zhang, Zhenguo Li, and James T. Kwok. Forward-Backward Reasoning in Large Language Models for Mathematical Verification, 2024. URL https://doi.org/10.18653/v1/2024.findings-acl.397.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield Dodds, Nova DasSarma, Eli Tran-Johnson, Scott G. Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack A. Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language Models (Mostly) Know What They Know, 2022. URL http://arxiv.org/abs/2207.05221.

Amita Kamath, Robin Jia, and Percy Liang. Selective Question Answering under Domain Shift, 2020. URL https://aclanthology.org/2020.acl-main.503/.

Leslie Kish. *Survey Sampling*. John Wiley & Sons, New York, 1965. ISBN 047148900X.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic Uncertainty: Linguistic Invariances for Uncertainty Estimation in Natural Language Generation, 2023. URL http://arxiv.org/abs/2302.09664.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, pp. 611–626, 2023. doi: 10.1145/3600006.3613165.

Jungseob Lee and Chanjun Park. The agent loop: A survey of control strategies, skills, and harnesses for llm agents. *SSRN working paper*, 2026. doi: 10.2139/ssrn.7186738. URL https://ssrn.com/abstract=7186738.

Jungseob Lee, Seongtae Hong, Seungjun Lee, Jaehyung Seo, Junyoung Son, Sugyeong Eo, Chanjun Park, Hyeongju Park, Hyeonseok Moon, and Heuiseok Lim. DART: Draft-Agreement Routing for Training-Free Adaptive Thinking Budgets in Hybrid Reasoning Models, 2026a. URL https://arxiv.org/abs/2606.23181.

Jungseob Lee, Seungyoon Lee, Suhyune Son, Dongyub Jude Lee, Sungbin Han, Sugyeong Eo, and Heuiseok Lim. Answer-Conditioned Chains of Thought Degrade Verifiable-Reasoning Distillation in Large Language Models, 2026b. URL https://arxiv.org/abs/2607.14552.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, pp. 3843–3857, 2022. doi: 10.52202/068431-0278.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. Escape Sky-high Cost: Early-stopping Self-Consistency for Multi-step Reasoning, 2024. URL http://arxiv.org/abs/2401.10480.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's Verify Step by Step, 2023. URL http://arxiv.org/abs/2305.20050.

Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. Deductive Verification of Chain-of-Thought Reasoning, 2023. URL https://doi.org/10.52202/075280-1580.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. Improve Mathematical Reasoning in Language Models by Automated Process Supervision, 2024. URL http://arxiv.org/abs/2406.06592.

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, 2023. URL https://doi.org/10.52202/075280-2019.

Potsawee Manakul, Adian Liusie, and Mark Gales. SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models, 2023. URL https://doi.org/10.18653/v1/2023.emnlp-main.557.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, pp. 124198–124235, 2024. doi: 10.52202/079017-3946.

Vy Nguyen, Ziqi Xu, Jeffrey Chan, Estrid He, Feng Xia, and Xiuzhen Zhang. Hallucinate Less by Thinking More: Aspect-Based Causal Abstention for Large Language Models, 2026. URL https://doi.org/10.1609/aaai.v40i38.40532.

Maxim Panov, Artem Shelmanov, Roman Vashurin, Artem Vazhentsev, Ekaterina Fadeeva, Lyudmila Rvanova, and Timothy Baldwin. Uncertainty Quantification for Large Language Models, 2026. URL https://doi.org/10.1007/978-3-032-21321-1_7.

Victor Quach, Adam Fisch, Tal Schuster, Adam Yala, J. Sohn, T. Jaakkola, and R. Barzilay. Conformal Language Modeling, 2023. URL https://arxiv.org/abs/2306.10193.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. ZeRO: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16, 2020. doi: 10.1109/SC41405.2020.00024.

J. N. K. Rao and A. J. Scott. The analysis of categorical data from complex sample surveys: Chi-squared tests for goodness of fit and independence in two-way tables. *Journal of the American Statistical Association*, 76(374):221–230, 1981. doi: 10.1080/01621459.1981.10477633.

Charlie Snell, Jae-Hoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters, 2024. URL http://arxiv.org/abs/2408.03314.

Kaya Stechly, Matthew Marquez, and Subbarao Kambhampati. GPT-4 Doesn't Know It's Wrong: An Analysis of Iterative Prompting for Reasoning Problems, 2023. URL http://arxiv.org/abs/2310.12397.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. Just Ask for Calibration: Strategies for Eliciting Calibrated Confidence Scores from Language Models Fine-Tuned with Human Feedback, 2023. URL https://doi.org/10.18653/v1/2023.emnlp-main.330.

Ryan J. Tibshirani, Rina Foygel Barber, Emmanuel J. Candès, and Aaditya Ramdas. Conformal Prediction Under Covariate Shift, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/8fb21ee7a2207526da55a679f0332de2-Abstract.html.

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, 2022. URL http://arxiv.org/abs/2211.14275.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, 2005. doi: 10.1007/b106715. URL https://doi.org/10.1007/b106715.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations, 2024. URL https://doi.org/10.18653/v1/2024.acl-long.510.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-Consistency Improves Chain of Thought Reasoning in Language Models, 2022. URL http://arxiv.org/abs/2203.11171.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large Language Models are Better Reasoners with Self-Verification, 2023. URL https://doi.org/10.18653/v1/2023.findings-emnlp.167.

Wei Xiong, Hanning Zhang, Nan Jiang, and Tong Zhang. An implementation of generative PRM. https://github.com/RLHFlow/RLHF-Reward-Modeling, 2024.

Tianci Xue, Ziqi Wang, Zhenhailong Wang, Chi Han, Pengfei Yu, and Heng Ji. RCOT: Detecting and Rectifying Factual Inconsistency in Reasoning by Reversing Chain-of-Thought, 2023. URL http://arxiv.org/abs/2305.11499.

Yasin Abbasi Yadkori, Ilja Kuzborskij, David Stutz, András György, Adam Fisch, Arnaud Doucet, Iuliya Beloshapka, Wei-Hung Weng, Yao-Yuan Yang, Csaba Szepesvári, Ali Taylan Cemgil, and Nenad Tomasev. Mitigating LLM Hallucinations via Conformal Abstention, 2024. URL http://arxiv.org/abs/2405.01563.

Yusuke Yamauchi, Taro Yano, and Masafumi Oyamada. An Empirical Study of LLM-as-a-Judge: How Design Choices Impact Evaluation Reliability, 2026. URL https://doi.org/10.18653/v1/2026.gem-main.19.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-Math Technical Report: Toward Mathematical Expert Model via Self-Improvement, 2024. URL http://arxiv.org/abs/2409.12122.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 Technical Report, 2025. URL https://arxiv.org/abs/2505.09388.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative Verifiers: Reward Modeling as Next-Token Prediction, 2024. URL http://arxiv.org/abs/2408.15240.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The Lessons of Developing Process Reward Models in Mathematical Reasoning, 2025. URL https://doi.org/10.18653/v1/2025.findings-acl.547.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena, 2023. URL https://doi.org/10.52202/075280-2020.

Bowen Zuo, Dongruo Zhou, and Yinglun Zhu. Adaptive Test-Time Compute Allocation with Evolving In-Context Demonstrations, 2026. URL https://doi.org/10.18653/v1/2026.findings-acl.1754.

A Implementation details

A.1 The frozen answer pool

Five generators, each a fine-tune of `Qwen/Qwen3-8B-Base`, attempt the same 500-problem subset (`HuggingFaceH4/MATH-500`, split `test`) (Hendrycks et al., 2021). Of the 2500 attempts, 2331 carry an extractable answer over 492 problems, a mean of 4.74 answers per problem. The 169 attempts that are missing are exactly those whose extracted answer came back empty, and 149 of those ran to the 32768-token generation cap. The pool holds 99 wrong answers, a base error of 4.25%, or 10.72% if the 169 unanswered attempts are counted as failures. The evaluation basis is 2326 answers with a base error of 4.26%, since 5 rows lack a complete vote bank and are excluded. All scorers evaluate the same stored candidate answers.

Table 4: Composition of the frozen pool per generator. Every dropped attempt has an empty extracted answer.

			Dropped attempts		
Generator	Evaluated	In pool	All	Empty answer	At token cap
iter2	500	470	30	30	26
SelfDistill	500	468	32	32	26
ForwardPref	500	467	33	33	30
STaR	500	464	36	36	32
F0	500	462	38	38	35
Total	2500	2331	169	169	149

A.2 THE FIVE GENERATORS

All five are trained under DeepSpeed ZeRO-3 with CPU offload at `max_seq_length` 8192, for one epoch except SelfDistill at three (Rajbhandari et al., 2020).

Table 5: The five pool sources and how each was built. All five start from the same 8B backbone.

Source	Construction
F0	Forward supervised fine-tune of the backbone on a 20k set
STaR	Three iterations of sample 8 per problem, filter, fine-tune 1 epoch, starting from F0
SelfDistill	Sample 8 per problem from F0, fine-tune from `Qwen/Qwen3-8B-Base`, 3 epochs
ForwardPref	Preference pairs from forward samples, SimPO-style trainer, $\beta = 2.0$, $\gamma = 1.0$, from F0 (Meng et al., 2024)
iter2	Iteration-2 checkpoint of the preference-training loop

A.3 THE BACKWARD PROBE

The reverse stage draws $n = 3$ samples for every production run at `max_tokens` 384, `temperature` 1.0, `top_p` 0.95, `logprobs` 1 and `repetition_penalty` 1.0. The trace we keep is the text before the end-of-thinking marker, and the reverse score is the mean token log-probability of that trace. The recovery stage uses its default style at `temperature` 0.0, `top_p` 1.0 and `max_tokens` 1536. The engine is vLLM at `max_model_len` 16384, `gpu_memory_utilization` 0.82, `max_num_seqs` 2 and `VLLM_USE_V1` 0. The observation m counts how many of the three recovered answers match the proposal under the canonical matcher.

Backward probe, the two prompts

Reverse.
```
Question: {problem}
Answer: {pred}
Generate the reasoning process that leads from the
question to this answer.
```

Recovery. the chat-templated question followed by `Put the final answer inside \boxed{}.` with the reverse trace prefilled as thinking and terminated by `</think>`.

There are three probe variants. They differ only in the checker and all three run on the same 8B pool. The 8B variant pairs an 8B reverse fine-tune of the backbone with `iter2` as the forward model. The 1.7B variant pairs a reverse adapter on `Qwen/Qwen3-1.7B` with `Qwen/Qwen3-1.7B`, and the 4B variant pairs a reverse adapter on `Qwen/Qwen3-4B` with `Qwen/Qwen3-4B`.

A.4 THE RE-SOLVE VOTE BANKS

For every unique problem we draw fresh solutions from the frozen forward model at the pool's own decoding settings, `n` 8, `max_tokens` 30720, `max_model_len` 32768, `temperature` 0.6, `top_p` 0.95, `top_k` 20, `presence_penalty` 0.4, `repetition_penalty` 1.0, `max_num_seqs` 24, `gpu_memory_utilization` 0.90 and thinking enabled. The 8B bank holds 16 votes per problem, in two independent batches of 8, and the 1.7B bank holds 8 votes per problem. Answer extraction reads a boxed answer from the text after the last end-of-thinking marker and falls back to the whole text. Offline, a candidate's vote vector records whether each draw matches the proposal, and vote banks are indexed by the full problem text. A bank described as boxing appends the instruction `Please reason step by step, and put your final answer within \boxed{}.` to the prompt.

A.5 GENERATOR CONFIDENCE ACTIONS

The first confidence action reads the generator's probability of Yes under the prompt below, with thinking disabled, `max_tokens` 1, `temperature` 0.0 and `logprobs` 20. The score is

$P(\text{Yes})/(P(\text{Yes})+P(\text{No}))$ with mass summed over every surface form that appears in the returned top-k, and it is missing for 13 of the 2331 rows. The second action is the exact mean token log-probability of the stored solution under the forward model. Youden-optimal thresholds on the full pool are 0.9241 for the probability of Yes, with 13 nulls, and -0.1688 for the mean log-probability, with none. We impute missing scores to the minimum observed score and retain every row. All declared thresholds exceed this imputation floor, so none accepts a missing-score row.

The confidence prompt, verbatim

```
"Question:\n{problem}\n\nProposed answer:  {pred}\n\nIs the
proposed answer correct?  Reply with Yes or No only."
```

A.6 Correctness matching and unextractable answers

There are two matchers and they are not interchangeable. The canonical analysis-time matcher normalises text macros, spacing, matrix environments, fraction forms, thousands separators and degree marks, rewrites fractions to a fixpoint, and then expands a variant set that canonicalises commutative addition, plus-minus sets, top-level comma sets, single-letter choices, equation right-hand sides, set membership, base subscripts, units and ordinals. A match is a variant-set intersection first and a numeric fallback at absolute tolerance 10^{-6} second, and nothing looser. An answer is labelled correct when any stored evaluation label or a direct canonical match of the proposal against the gold answer marks it correct. On the second benchmarks, the stored labels also come from a generation-time matcher that tries a symbolic parser first, a numeric fallback at relative tolerance 0.02 second and the canonical matcher third. Coverage there is identical under four alternative labellings, since no accept rule reads the gold answer.

An attempt with no extractable answer is not in the pool at all, so selective risk is unaffected and only the base error reading moves. An empty draw inside a vote bank counts as not agreeing. In the controller replay we give each empty extraction its own singleton cluster label, so that empties are neither dropped from the denominator nor allowed to gang up into a plurality.

A.7 Cost accounting on three axes

The first unit is generated tokens. The second is generated tokens weighted by parameter count against the 8B reference, with weights 1.0 for 8B, 0.5 for 4B and 0.21 for 1.7B, where $0.21 = 1.72\text{B}/8.19\text{B}$ from the released parameter counts. These two token proxies count generated text and exclude the cost of processing input prompts.

Certification charges each three-draw probe a fixed 3400 tokens per candidate. The pooled cost tables use a per-row estimate: reverse-trace characters converted at 0.3170 tokens/character, plus a fixed recovery mean of 1984 tokens. This estimate averages 3101 tokens on the 2326 evaluation answers. Re-solve cost is the number of draws used times the stored thinking-token count plus 400 per draw. Cascade cost is additive over the stages actually executed, so a first-stage rejection pays for that stage alone. We also average cost over all rows rather than over kept rows, so an abstention still pays for the checks it ran.

The third axis estimates accelerator time from throughput measured on one A100-SXM4-80GB with one model alone, at `max_num_seqs` 16, `max_model_len` 16384, `gpu_memory_utilization` 0.9, `temperature` 0.6, `top_p` 0.95, `top_k` 20 and `presence_penalty` 0.4. Every latency number is measured with one model alone on one device. The probe timing harness uses a 384-token question-only trace stage and a 512-token recovery stage with the same model. These timing prompts differ from the production probe in Appendix A.3. We scale the measured seconds/token by stored action and policy token counts to estimate their accelerator time.

A.8 Schedule families and the deployed grid

The sequential primitive accepts at k agreements, rejects at j disagreements and rejects on exhaustion. We build seven families over it. A threshold family accepts when the probe agreement count reaches $\tau \in \{2, 3\}$, at 8B, 4B or 1.7B. A unanimity family accepts when the first $N \in \{1, 2, 3, 4\}$

8B votes agree, with one 1.7B member at $N = 1$. An adaptive family runs the race on the 8B bank at $k \in \{2, 3\}$ and $j \in \{2, 3\}$.

A probe-first cascade accepts at three probe agreements, rejects at zero, and sends the one-to-two band to the race. A score-gated variant accepts fast at two probe agreements when the reverse score is at least -0.30. A small-first cascade accepts on three of three 1.7B votes, rejects at zero, and sends the band to the 8B race, and its probe-first sibling puts the 1.7B probe in the first stage. A triple cascade runs the 1.7B probe, then a 1.7B adaptive vote, and then a single confirming 8B solve.

The deployed grid holds 23 configurations. Its small-first cascade accepts on three of three 1.7B votes, rejects on zero agreements, and races the 8B bank in the one-to-two band, needing two agreements before two or three refusals. The small-first cascade in the pooled tables of this appendix races to three agreements before two refusals, a variant outside the deployed grid.

A.9 Certification constants, family levels and split sizes

The confidence budget is $\delta = 0.05$ throughout, and the minimum calibration coverage the cheapest-clearing selector may consider is 0.60. The targets we sweep are $\alpha \in \{0.005, 0.0075, 0.010, 0.0125, 0.015, 0.020\}$ with primary 0.015, chosen by the nested-protocol criterion in Section 3.3.

The Bonferroni thresholds are $0.05/23 = 0.00217$ for the policy grid, $0.05/24 = 0.00208$ for the step-level threshold families and $0.05/6 = 0.00833$ for the outcome-level scorers. The minimum zero-error kept count that can certify is $\lceil \log(\delta/|G|)/\log(1-\alpha) \rceil$.

The multiplicity comparison in Figure 4b uses a 20-member family at $0.05/20 = 0.0025$. Its corrected arm attains the deployed grid's 76.09% at the primary target.

The rival families of Table 2 pre-specify quantile levels, since absolute thresholds are not comparable across scorers on different scales. The joint family is 4 aggregations by 6 levels, 0.02, 0.05, 0.10, 0.20, 0.30 and 0.50, for 24 rules, and the single-aggregation family is 1 aggregation by 24 levels from 0.02 to 0.48. Threshold locations use calibration scores but not labels. These score-adaptive families are compared under the same nominal answer-level test; score-only construction does not supply the fixed-rule independence condition of Section 3.3.

The split sizes are as follows. The five source-held-out folds carry calibration sides of 1857 to 1865 and test sides of 461 to 469. The ten question-disjoint halves carry calibration sides of 1150 to 1169 and test sides of 1157 to 1176, from ten random problem partitions.

A.10 Composition and prediction machinery

Per class the fitted rate is $p = \mathrm{clip}(\mathrm{mean}(\mathrm{counts})/K, 10^{-4}, 1 - 10^{-4})$ and the overdispersion is $\rho = (\mathrm{var}(\mathrm{counts})/(Kp(1-p)) - 1)/(K-1)$ clipped to $[0, 0.95]$, and we set $\rho = 0$ when fewer than five rows are available. Prediction is binomial when $\rho < 0.01$ and Beta-Binomial otherwise, with $a = p(1-\rho)/\rho$ and $b = (1-p)(1-\rho)/\rho$. That gives predicted coverage $\pi\,\mathrm{tail}_1 + (1-\pi)\,\mathrm{tail}_0$ and predicted risk $(1-\pi)\,\mathrm{tail}_0/\mathrm{pred_cov}$.

The sequential accept probability for the adaptive race is the memoised recursion $f(k, j) = pf(k-1, j) + (1-p)f(k, j-1)$ with $f(0, \cdot) = 1$ and $f(\cdot, 0) = 0$. We compose a two-stage cascade from single-action prices alone as

$$\mathrm{acc}[y] \;=\; \mathrm{pmf}(3, 3, p_y, \rho_y) \;+\; \big(\mathrm{pmf}(3, 1, p_y, \rho_y) + \mathrm{pmf}(3, 2, p_y, \rho_y)\big)\, f\big(2, 2 \,\big|\, p_y^v\big). \tag{5}$$

We stratify the probe by a source quota with a wrong-class floor of 8. When a draw carries fewer than eight wrong answers, wrong answers are swapped in from the remainder and an equal number of correct ones are dropped. The 100-answer fitting set therefore requires labels beyond those retained in it. We use its correct-answer fraction as the plug-in prior: on MATH this averages 0.9199, against 0.9574 in the pool. The prediction errors evaluate this complete sampling and fitting recipe.

A.11 Compute, runtime and environment

The verification path is offline re-scoring of stored banks, and certification itself is deterministic and runs on CPU. We ran the supervised scorers on two RTX A6000 48GB devices, one scorer per

device, with transformers 4.53.1. The verification pipeline runs at `max_model_len` 16384, and the step-level reward model runs at a 4096-token cap, its maximum position count.

B BASELINE CONSTRUCTION

All baselines are row-aligned with the frozen pool and use the same canonical correctness labels. The four reward-model and prompted-judge scorers below read the presented solution with its thinking trace stripped; the classifier's input is specified separately. All scorers cover the same 2331 candidate answers, and the common evaluation basis contains the 2326 with complete vote banks.

B.1 THE FOUR TRAINED AND PROMPTED SCORERS

The step-level process reward model is `Qwen/Qwen2.5-Math-PRM-7B` (Yang et al., 2024). We split the solution on blank lines into steps and insert a separator token, and the step reward is the positive-class probability at each separator, at a 4096-token cap. We record four answer-level aggregations, minimum, product, mean and last, with the product as the default.

Math-Shepherd is `peiyi9979/math-shepherd-mistral-7b-prm`, and we run it under the official card protocol verbatim (Wang et al., 2024). Its good and bad tokens, step tag and plain-text input follow the card, at a 4096-token cap.

The outcome reward model is `RLHFlow/Llama3.1-8B-ORM-Deepseek-Data` (Xiong et al., 2024). The readout is a softmax over the plus and minus logits at the assistant turn. Because there is one scalar per answer, all four aggregation columns carry the same value and the step count is one.

The label-free judge is `Qwen/Qwen3-8B`, the instruction-tuned release of the exact family and size the five generators are built on (Yang et al., 2025). We use the instruction-tuned release because the base checkpoint has no chat template. The score is $P(\text{Yes})/(P(\text{Yes}) + P(\text{No}))$ over the first generated verdict token with mass summed over every surface form in the returned top-k. We evaluate four prompt variants and pick the primary by ranking area under the curve. The direct variant asks a one-line binary question, the rubric variant adds a grader persona and three criteria, the chain-of-thought variant verifies in the open and then forces a verdict, and the thinking variant uses the model's native thinking mode and then forces a verdict.

The two rationale variants score in two passes. The first generates the rationale at temperature 0.0 when a single sample is drawn and 0.7 with `top_p` 0.95 otherwise, at 512 rationale tokens for the chain-of-thought variant and 1024 for the thinking variant. The second re-feeds prompt, rationale and a verdict cue and reads the next-token distribution. The presented solution is capped at 4096 tokens with the middle elided, so the final boxed answer always survives, and we carry the thinking variant forward.

The classifier trained on the certificate's own labels is `Qwen/Qwen3-1.7B` used as a two-label sequence classifier in bfloat16 with gradient checkpointing, one epoch, AdamW at learning rate 10^{-5}, gradient-norm clipping at 1.0 and inverse-frequency class weights, since the pool runs about 24 correct to 1 wrong. The batch size is 4. The input renders the problem, the proposed answer and the trace, with the head capped at half the context and the tail of the trace kept.

We partition calibration into four folds and score each fold by a model trained on the other three, so no row is certified by a model that trained on it. The test side is scored by the ensemble of the four.

The certification family has 24 thresholds, matching the reward model's family size and correction. We evaluate three threshold families: one carrying calibration levels to test, one re-evaluating levels on the test side, and one spreading 24 absolute cut points over the unit interval.

B.2 Score resolution, and the divisor a family actually pays

Table 6: Score resolution and certification by scorer: wrong answers kept at matched coverage, ranking AUC and distinct score values by aggregation, the single-aggregation family at 1.5%, and what the 24-rule grid realises over splits. Bold marks a step-level scorer's best aggregation.

	Step-level scorers		One scalar per answer			
	PRM	Shepherd	ORM	Judge	$p(\text{True})$	Log-prob.
Wrong kept at matched coverage	97	164	195	60	372	295
Ranking AUC by aggregation						
Product	0.8752	0.7634	0.8113	0.9231	0.5572	0.7050
Minimum	0.8574	0.7820	0.8113	0.9231	0.5572	0.7050
Mean	**0.8861**	**0.8156**	0.8113	0.9231	0.5572	0.7050
Last step	0.7565	0.8113	0.8113	0.9231	0.5572	0.7050
Best minus product	+0.0109	+0.0522	0.0000	0.0000	0.0000	0.0000
Distinct score values by aggregation						
Product	1716	2311	86	933	269	2326
Minimum	545	57	86	933	269	2326
Mean	1316	1786	86	933	269	2326
Last step	64	54	86	933	269	2326
Single-aggregation family of 24 levels, at the 1.5% target						
Splits certifying	15	2	2	15	0	0
Coverage, all splits (%)	66.55	7.61	7.66	71.91	0.00	0.00
Realised by the 24-rule grid, range over splits						
Distinct thresholds	22	20–24	6	6	6	6
Distinct accept sets	9–13	14–19	6	6	6	6

Table 6's single-aggregation rows fix the aggregation that certifies the most coverage, the mean for the reward model and the last step for Shepherd, and sweep 24 levels. Their coverage averages all 15 splits, a split that certifies nothing counting as zero.

Table 6 shows coarse score resolution for the outcome model and Shepherd's minimum and last-step aggregations, which use a softmax over two bfloat16 logits. Its last two rows explain the different divisors. A scorer that returns one scalar per answer realises six distinct accept sets from a nominal 24-rule grid, since its four aggregation columns are identical before any data is read, so it pays $0.05/6$ and not $0.05/24$. A step-level scorer realises between 9 and 19 and pays $0.05/24$.

B.3 The permutation protocol behind the controller rows

Adaptive consistency and early-stopping consistency are sequential, so their cost depends on draw order. We therefore average every sequential number over 1000 independent random permutations of each problem's eight draws over 2326 evaluation rows. Table 19 reports these permutation means. We run self-consistency at fixed N over the whole curve from 1 to 8 draws, with a tie-abstaining variant, since first-seen tie-breaking, which is what both official implementations do, makes two draws identical to one. Adaptive consistency uses the official stopping criterion evaluated after every draw including the first, as the official evaluation loop does, with its threshold swept from 0.50 to 0.99. The early-stopping controller uses windows of 2, 3 and 4 at a budget of 8, and the official loop takes no remainder window. None of the three controllers carries a risk guarantee, so we report their realised operating points in Table 19.

B.4 THE CLASSIFIER TRAINED ON THE CERTIFICATE'S OWN LABELS, BY CONTEXT WINDOW

Table 7: A 1.7B correctness classifier over four context windows, with four-fold cross-fitting, a four-model test ensemble and the identical protocol at family size 24. Coverage at the 1.5% target averages all 15 splits, a split that certifies nothing counting as zero.

		Out-of-fold AUC		At the 1.5% target		Matched
Window	Input	Mean	Range	Splits	Cov. (%)	Wrong
1024	Raw text	0.7284	0.5713–0.8131	0/15	0.00	225
2048	Presented	0.7779	0.5793–0.8627	2/15	4.54	185
4096	Raw text	0.8029	0.6730–0.8872	2/15	7.73	147
8192	Raw text	0.8229	0.7088–0.8915	3/15	8.40	123

Table 7 uses the quantile-matched family and keeps 10504 answers at matched coverage in every window; PriceCheck keeps 44 wrong at that total. At the widest window the absolute-threshold family certifies on 0 of 15 splits, with a median first-certifying target of 0.075, and the quantile-matched family certifies on 3 of 15, with a median of 0.03.

C CERTIFICATION RESULTS ACROSS TARGETS AND SPLITS

C.1 MODAL SELECTIONS BY SCHEME, TARGET AND SELECTOR

Table 8: Certification results by split scheme, risk target and selector. Each row gives the modal selected configuration, the splits selecting it and its held-out means over them, for a 23-configuration grid on 2326 answers. No question-disjoint half certifies at 0.5%.

				Held-out (%)		Tokens per answer	
Target	Selector	Modal configuration	Splits	Cov.	Risk	Generated	Weighted
Source-held-out folds							
0.5%	Max. coverage	Unanimous 4-vote, 8B	5/5	69.6	0.00	21533	21533
	Min. cost	Unanimous 4-vote, 8B	5/5	69.6	0.00	21533	21533
0.75%	Max. coverage	Unanimous 4-vote, 8B	5/5	69.6	0.00	21533	21533
	Min. cost	Unanimous 4-vote, 8B	5/5	69.6	0.00	21533	21533
1.0%	Max. coverage	Unanimous 3-vote, 8B	5/5	71.6	0.18	16150	16150
	Min. cost	Unanimous 3-vote, 8B	5/5	71.6	0.18	16150	16150
1.25%	Max. coverage	Adaptive vote (strict)	2/5	78.5	0.55	17232	17232
	Min. cost	Cascade, score fast path	2/5	71.0	0.61	4810	4810
1.5%	Max. coverage	Adaptive vote, 8B	5/5	79.8	0.54	13426	13426
	Min. cost	Small-first cascade	3/5	77.7	0.55	15558	4107
2.0%	Max. coverage	Adaptive vote, 8B	5/5	79.8	0.54	13426	13426
	Min. cost	Backward probe, 1.7B	5/5	63.2	0.61	3400	714
Question-disjoint halves							
0.75%	Max. coverage	Unanimous 4-vote, 8B	3/10	68.1	0.00	21550	21550
	Min. cost	Unanimous 4-vote, 8B	3/10	68.1	0.00	21550	21550
1.0%	Max. coverage	Unanimous 3-vote, 8B	4/10	72.5	0.36	16104	16104
	Min. cost	Unanimous 3-vote, 8B	4/10	72.5	0.36	16104	16104
1.25%	Max. coverage	Unanimous 3-vote, 8B	4/10	72.5	0.36	16104	16104
	Min. cost	Unanimous 3-vote, 8B	4/10	72.5	0.36	16104	16104
1.5%	Max. coverage	Unanimous 3-vote, 8B	7/10	71.5	0.20	16129	16129
	Min. cost	Unanimous 3-vote, 8B	7/10	71.5	0.20	16129	16129
2.0%	Max. coverage	Adaptive vote, 8B	6/10	81.0	0.62	13727	13727
	Min. cost	Unanimous 2-vote, 8B	3/10	75.2	0.34	10643	10643

Table 8 reports modal selections. Across all configurations, the certification counts over 15 splits show that at a 0.5% target only the unanimous 4-vote certifies, on 5 of 15, and everything else on none. At 1% the unanimous 4-vote certifies on 15 of 15, the unanimous 3-vote on 10 of 15 and every other configuration on at most 2 of 15. At 1.5% the two unanimous votes certify on 15 of 15 and every other configuration on 1 to 8 of 15. At 2% nothing is below 7 of 15 and the two unanimous votes remain at 15 of 15.

Table 8 evaluates 2326 answers with a base error of 4.26%. The population coverage ceiling from Section 3.3 is therefore 97.20% at the primary 1.5% target.

C.2 THE POLICY THE CERTIFICATE SELECTS ON EACH SPLIT

Table 9: The certified selection on each of the 15 shared splits at the 1.5% target, with the number of grid members certifying there, the held-out result and the calibration and test sizes. Race (k, j) serves at k agreements before j disagreements; A–J identify the ten problem partitions.

			Held-out			Answers	
Split	Selected configuration	Certified	Cov. (%)	Risk (%)	Weighted tok.	Calibration	Test
Source-held-out folds, by held-out generator							
F0	Race $(2, 3)$	19	79.0	0.27	13605	1865	461
ForwardPref	Race $(2, 3)$	19	80.7	0.53	13145	1860	466
STaR	Race $(2, 3)$	16	80.6	0.54	12927	1863	463
SelfDistill	Race $(2, 3)$	18	79.0	0.81	12929	1859	467
iter2	Race $(2, 3)$	16	80.0	0.53	14524	1857	469
Question-disjoint halves							
A	Unanimous 3-vote	2	71.8	0.36	16522	1169	1157
B	Unanimous 3-vote	2	73.4	0.35	15481	1150	1176
C	Unanimous 3-vote	2	71.5	0.36	17809	1165	1161
D	Race $(2, 3)$	22	80.9	0.75	13896	1167	1159
E	Unanimous 3-vote	2	68.5	0.00	16601	1153	1173
F	Race $(2, 3)$	22	81.1	1.06	14184	1159	1167
G	Unanimous 3-vote	2	70.3	0.00	16127	1164	1162
H	Unanimous 3-vote	2	71.7	0.00	15760	1163	1163
I	Race $(3, 3)$	11	79.6	0.54	17305	1157	1169
J	Unanimous 3-vote	2	73.3	0.35	14603	1161	1165

C.3 RE-SOLVE AGREEMENT AS A SCALAR SCORE

Here we score each answer by the fraction of its first n stored re-solves that agree with it and certify that score exactly as Table 2 certifies a scalar rival, with six quantile levels. We charge n times the answer's stored solve length, as the grid charges a vote.

Table 10: Re-solve agreement fractions certified under the shared protocol of the main comparison, with generated tokens per answer. Coverage at the 1.5% target averages all 15 splits, a split that certifies nothing counting as zero, and wrong answers are kept at the schedule's coverage.

		Splits certifying, of 15			At 1.5%	Matched
Score	Tokens	0.5%	1.0%	1.5%	Cov. (%)	Wrong
PriceCheck (ours)	15028	5	15	15	76.09	44
8B re-solve agreement						
2 re-solves	10768	0	1	8	39.73	58
3 re-solves	16152	0	10	15	72.23	55
4 re-solves	21536	5	15	15	73.61	32
8 re-solves	43071	5	15	15	75.67	30
16 re-solves	86142	5	15	15	76.23	28
1.7B re-solve agreement						
2 re-solves	9602	0	2	7	34.90	80
4 re-solves	19205	0	4	8	39.60	58
8 re-solves	38410	5	15	15	72.07	53

Table 10 shows that three 8B re-solves generate about as many tokens as the certified schedule, certify on fewer splits at the two strictest targets and serve less at 1.5%. Four or more keep fewer wrong answers at matched coverage, and sixteen, at about 5.7 times the schedule's tokens, reach its coverage. A family of 24 levels in place of the six quantile levels leaves this unchanged, since again only sixteen 8B re-solves reach the schedule's coverage.

C.4 A NESTED PROTOCOL OVER TWENTY REPEATS

Table 3b uses a nested protocol. The grid is re-selected on a design third of the problems from an 87-configuration space. For each of the targets 0.5, 1.0, 1.5 and 2.0%, the rule takes the five highest-coverage configurations whose one-sided 95% Clopper-Pearson bound on the design third is at most the target, together with the three cheapest of those whose design coverage is at least 0.60, and it keeps the first 20 distinct members. Learn-then-Test then runs on a calibration third, and we report the untouched test third, over 20 repeats. The realised grid size averages 16.1 with a standard deviation of 3.37, a minimum of 10 and a maximum of 20, and the mean calibration size is 693.5. The reward-model arm draws its family from the 508-rule survey by the same rule without the cost tier, which yields 13 to 16 rules in each repeat.

Table 3a uses two fixed reward-model families. The first crosses the four aggregations with six round absolute thresholds, 0.01, 0.05, 0.1, 0.2, 0.5 and 0.9, set before any score was read. The second surveys 508 rules, 200 quantiles of each aggregation's scores on the whole pool with duplicates removed, and keeps for each of those same four targets the six highest-coverage rules whose bound on the whole pool is at most the target, filling the rest of the 24 with the highest-coverage rules whose bound is at most 2%.

Table 11: The nested design, calibration and test protocol over 20 repeats, against the maximum-coverage selection on the source-held-out folds. Coverage is a percentage of the test third averaged over the certifying splits, and no split certifies at 0.5%.

	Nested protocol, 20 repeats		Source-held-out
Target	Splits certifying	Cov. (%)	Cov. (%)
1.0%	5/20	79.3	71.6
1.5%	10/20	76.3	79.8
2.0%	13/20	78.2	79.8

Table 11 shows no certification at a 0.5% target, since the minimum zero-error kept count exceeds the calibration third in every repeat. Over the 56 certified picks across targets and selectors, the test third reads above the target in 9, and 2 of these exceed it significantly at the 0.05 level against 2.8 expected by chance. At the primary target the ten certified splits pool to a test risk of 0.63%.

C.5 FIXED-SEQUENCE ORDERINGS AND THEIR LICENSING RATES

Fixed-sequence testing requires an order chosen independently of calibration. We test orders that share no problem with calibration: a price fitted on a probe whose problems are removed from both sides, a price fitted on a different benchmark, and three cheapest-action-first orders computed from policy definitions and published constants alone.

Figure 6 reports the share of the 300 cells, 20 redraws by 15 splits with one answer per problem in calibration, in which each order licenses a policy. At the primary target, flat Bonferroni at the deployed divisor licenses none of these cells, while the two price orders license 23 and 22 of 300, and no order licenses any cell at a 0.5% target.

D DEPENDENCE BETWEEN ANSWERS TO THE SAME PROBLEM

The pool holds 4.74 answers per problem, and the one-way analysis-of-variance intraclass correlation of its error indicator over 492 problems is 0.45. Answers to one problem are therefore not independent draws. We compare three nominal 95% upper bounds on selective risk. The answer-level bound applies Clopper-Pearson to kept answers, which is exact under independence. The one-per-problem bound keeps one answer per problem and averages the exact bound over 5000 redraws. The problem-count bound divides the kept count and the kept error count by the number of kept answers per kept problem, which reads no label, and takes the continuous extension of the binomial tail at the deflated counts. We assess these constructions in the simulations below; this assessment does not establish a general guarantee for clustered answers.

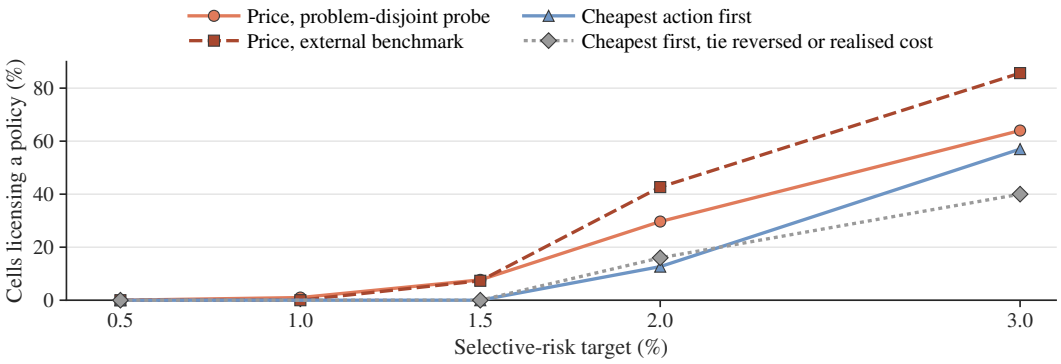

Figure 6: Share of 300 calibration cells, 20 redraws by 15 splits with one answer per problem, in which each pre-specified order licenses any policy, by selective-risk target.

D.1 Validity of each bound in simulation

On a beta-binomial matched to the pool, the one-per-problem bound keeps its nominal 95% coverage at every true risk from 0.55% to 5%, whereas the answer-level bound falls short of it as the correlation rises.

We check the level of the problem-count construction at the least favourable point of the null, where answer-weighted risk equals the target. A calibration side of 490 problems draws each problem's kept count from a policy's empirical kept counts on the source-held-out calibration sides, and it draws kept errors from a beta-binomial whose correlation is 0.45, 0.8 or 1.0, at targets of 1.5% and 2%, for the adaptive vote, the unanimous 3-vote and the small-first cascade, over 200,000 replications each. The problem-count construction rejects a true null at most 3.6% of the time at a nominal 5% and at most 0.12% at the Bonferroni level of 0.22%. Tilting the error rate toward problems that keep few answers, or many, at fixed answer-weighted risk leaves it within 3.2% and 0.07%.

D.2 The bound for every bounded policy

Table 12: Nominal 95% upper bounds on selective risk in percent for eleven policies under three constructions, with the kept answers, kept errors and distinct kept problems each is computed from.

	Kept			Upper bound on risk (%)		
Policy	Answers	Wrong	Problems	Answer level	One per problem	Problem count
At matched coverage						
Generator $p(\text{True})$	1724	62	373	4.424	6.625	5.618
Generator log-probability	1724	48	446	3.528	5.176	4.437
Process reward model	1724	16	418	1.406	2.462	2.137
Backward probe, 8B	1724	9	391	0.909	1.753	1.617
At each policy's own operating point						
Backward probe $m \geq 2$, 1.7B	1469	9	350	1.067	2.041	1.849
Backward probe $m \geq 2$, 8B	1589	10	361	1.065	1.949	1.844
Unanimous 3-vote, 8B	1665	3	341	0.465	1.267	1.197
Single re-solve, 8B	1776	12	371	1.092	1.960	1.887
Single re-solve, 1.7B	1799	13	378	1.146	2.195	1.938
Adaptive vote, 8B	1857	10	390	0.912	1.921	1.643
Small-first cascade	1798	7	375	0.730	1.441	1.452

Table 12 shows that counting each kept problem once widens every bound by 1.26 to 2.58 times and leaves the ordering of the eleven policies exactly as the answer-level column has it. The cascade's bound sits 0.69 points below the trained reward model's, at 1.45% against 2.14%. The problem-count bound is tighter than the one-per-problem mean on ten of the eleven rows, and on the cascade the two agree to 0.01 points.

E THE PRICE AND ITS TRANSPORT

E.1 PREDICTION ERROR AGAINST PROBE SIZE, FIVE POOLS

Table 13: Coverage prediction error in points and wrong-count prediction error in answers, at probe sizes of 50, 100 and 200 labels, on five pools. The realised coverage and wrong count of the predicted operating point are given for each pool.

		Realised		Coverage error			Wrong-count error		
Pool	Base acc. (%)	Cov. (%)	Wrong	50	100	200	50	100	200
MATH	95.8	68.2	10	5.07	3.60	2.55	15.14	10.87	7.83
GSM8K	90.4	89.2	7	3.26	2.07	1.35	7.31	4.36	2.61
Minerva	62.5	46.0	21	4.67	2.74	1.43	8.23	4.66	3.25
Qwen3-4B	95.2	81.8	2	4.52	2.80	1.68	3.59	3.02	2.11
Qwen3-1.7B	92.4	71.0	2	4.43	3.44	2.61	3.27	2.45	1.83

E.2 PREDICTED AGAINST REALISED COVERAGE ON THREE POOLS

Table 14: Coverage predicted from the two fitted rates against coverage realised on three answer pools, from a probe of 100 labels with a wrong-class floor of 8 over 20 redraws. Single-action columns give mean absolute error in points, and cascade columns give predicted against realised coverage.

	Mean absolute error (points)				Cascade (%)		Small-first (%)	
Pool	8B probe	8B vote	1.7B vote	All	Pred.	Real.	Pred.	Real.
MATH, $n = 2326$	3.46	3.95	3.49	3.64	70.8	71.1	76.0	77.8
GSM8K, $n = 500$	2.87	3.14	3.15	3.06	90.9	89.1	90.0	89.0
Minerva, $n = 272$	4.40	3.91	5.43	4.58	48.2	51.8	45.1	50.8

E.3 THE SAME PRICES ON FOUR ANSWER POPULATIONS

Table 15: Completeness and leak in percent for every action measured on every population, with the bank each rate is read from. All rates are per draw.

Action	p_1 (%)	p_0 (%)	Bank
MATH			
Backward probe, 8B	70.1	10.8	8B probe at three samples
Backward probe, 4B	73.1	13.8	4B probe at three samples
Backward probe, 1.7B	63.9	12.5	1.7B probe at three samples
Re-solve vote, 8B	80.2	9.0	8B vote bank
Re-solve vote, 1.7B	79.5	10.6	1.7B vote bank
Minerva			
Backward probe, 8B	72.9	30.5	Minerva probe
Re-solve vote, 8B	84.2	43.5	Minerva 8B vote bank
Re-solve vote, 1.7B	72.0	28.0	Minerva 1.7B vote bank
GSM8K			
Backward probe, 8B	97.6	16.3	500-row probe subset
Re-solve vote, 8B	94.0	23.3	1319-row vote bank
Re-solve vote, 1.7B	93.9	22.3	1319-row vote bank
Programs			
Backward probe, 8B	97.4	2.2	Pooled program candidates
Regenerate, 8B	75.4	3.1	Pooled program candidates

Table 15 reports per-action rates; the per-pool fits at three reverse samples are separate quantities. MATH has 2331 answers at base accuracy 95.8% with completeness 69.9, leak 10.8 and overdispersion 0.72 on the correct class and 0.59 on the wrong one. GSM8K has 500 answers at 90.4% with 97.0 and 16.7, Minerva 272 at 62.5% with 58.2 and 24.2, the 4B pool 499 at 95.2% with 83.0 and 12.5, and the 1.7B pool 497 at 92.4% with 72.5 and 10.5.

E.4 Transport to a second model lineage

On a pool of 280 answers from `meta-llama/Llama-3.1-8B-Instruct` at 38.9% accuracy, with three reverse samples, a probe of 100 labels, a wrong-class floor of 8 and 20 draws, the single-probe price predicts coverage at thresholds of one, two and three agreements with a mean error of 3.6 points over 60 comparisons. This transport experiment evaluates the single probe.

E.5 Further composition diagnostics over the 118-policy grid

Table 16 reports prediction errors from 20 redraws of the 100-answer class-enriched fitting set, evaluated on the remaining 2226 answers in the frozen pool. Problems can occur on both sides; the comparison measures prediction on an answer-level complement. Certification uses the separate 23-member family.

Table 16: Prediction quality over 70 cascades and all 118 schedules. Errors are mean absolute values.

Quantity	70 cascades	118 schedules
Coverage error (points)	2.97	3.86
Coverage rank correlation	0.99	0.97
Risk error (points)	0.97	0.95
Risk bias (points)	0.50	0.52

Table 16 summarises cascade prediction quality. Within this grid, coverage error falls by 1.36 points with each additional action in a chain. Across the 70 cascades, the mean signed relative cost biases are -3.0% on generated tokens and -4.9% on parameter-weighted tokens.

Table 16 uses the 100-answer fitting recipe. A full-pool fit gives a mean coverage error of 0.25 points across nine single-action thresholds, three thresholds for each of the 8B probe, 8B vote and 1.7B vote. Under that fit, the separately evaluated probe-first and small-first cascades retain errors of 2.76 and 0.85 points, respectively.

Table 16 uses the factorised composition rule, which multiplies the first stage's band probability by the second stage's acceptance over every answer of a class. This assumes independence given the label. Over the 68 two-stage cascades of the grid, a correct answer that the first stage leaves undecided is accepted by the 8B race 23.0 points less often than a correct answer overall, and a wrong one 5.4 points more often.

Table 16 provides a scale for the effect of this assumption. On the whole pool, the factorised and exact compositions share every other quantity. Replacing the former with the latter moves composed coverage by 0.92 points on average, with a problem-level bootstrap interval of 0.67 to 1.20 points and a worst cascade at 2.24, and it moves composed risk by 0.05 points, with no cascade's interval excluding zero. That is a third of the grid's 2.97-point coverage error and about a twentieth of its 0.97-point risk error.

F The three cost axes, per action and per policy

Table 17: Measured throughput and the resulting action-time estimates on one A100 80GB device. Seconds/action scales the measured seconds/1k tokens to each action's token count. Latency percentiles are measured completion times for 100 to 400 requests submitted together at 16 concurrent sequences, including queueing. Timing-run tokens are averaged over checked answers.

	Seconds		Latency (s)		
Action	/1k tokens	/action	p50	p99	Timing-run tokens/answer
Backward probe					
1.7B	0.545	0.563	41.2	86.8	2688
4B	0.782	0.808	59.2	124.5	2688
8B	1.079	1.115	82.1	172.4	2688
Re-solve					
1.7B	0.838	4.022	161.9	332.4	3968
8B	1.515	8.154	736.1	1235.0	8155

Table 17 shows the small model's advantage in measured token-processing rates. The parameter ratio is 0.21 for 1.7B against 8B, while the measured seconds/token ratio is 0.51 for a backward

probe and 0.55 for a re-solve. The proxy therefore overstates that rate advantage by 2.4 and 2.6 times. For the 4B model the proxy ratio is 0.49 against a measured 0.72. The two axes also order the actions differently, since the proxy places a 1.7B re-solve before an 8B probe and the throughput-based estimate reverses them.

Table 18: Policy costs on three axes. Accelerator time is estimated from measured throughput and the token total split by which model generated it.

	Tokens per answer				
Policy	Generated	Weighted	By 1.7B	By 8B	Est. seconds
Small-first cascade	15744	4365	14404	1340	14.096
Adaptive vote, 8B	13427	13427	0	13427	20.335
Single re-solve, 1.7B	4801	1008	4801	0	4.022
Single re-solve, 8B	5384	5384	0	5384	8.154

Table 18 gives each mixed policy's token split, derived from its raw and weighted totals at the fixed parameter weight using stored per-row 1.7B token counts.

Table 19: Operating point and cost of each policy on the frozen pool of 2326 answers with 99 wrong. Controller rows average 1000 draw permutations.

	Operating point			Tokens per answer	
Policy	Cov. (%)	Wrong	Risk (%)	Generated	Weighted
Single actions					
Backward probe $m \geq 1$, 8B	75.8	17	0.96	3101	3101
Backward probe $m \geq 2$, 8B	68.3	10	0.63	3101	3101
Backward probe $m \geq 2$, 4B	71.3	12	0.72	3101	1550
Single re-solve, 1.7B	77.3	13	0.72	4801	1008
Single re-solve, 8B	76.4	12	0.68	5384	5384
Schedules					
Small-first cascade	77.3	7	0.39	15744	4365
Adaptive vote, 8B	79.8	10	0.54	13427	13427
Unanimous 3-vote, 8B	71.6	3	0.18	16152	16152
Unanimous 4-vote, 8B	69.6	0	0.00	21536	21536
Consistency controllers					
Self-consistency, $N = 1$	77.3	8.8	0.49	5384	5384
Self-consistency, $N = 8$	80.4	11.2	0.60	43071	43071
Adaptive consistency, 0.8	80.3	11.1	0.60	18873	18873
Early stopping, window 3	80.2	10.1	0.54	22252	22252

G CONTROLS ON THE BACKWARD PROBE

G.1 DESTROYING THE PAIRING BETWEEN AN ANSWER AND ITS EVIDENCE

Table 20 reports controls that re-pair saved lists of recovered answers at random across problems, 1000 times in each cell, and recompute the match-count accept rule. No new recovery is generated. Actual coverage and precision are the unshuffled values on the same rows.

Table 20: Coverage when the pairing between an answer and its evidence is destroyed, by generator and agreement threshold, over 1000 trials. The last three columns are the mean, maximum and wrong-answer mean of the number of candidate answers accepted after re-pairing.

		Actual (%)		Shuffled coverage (%)			Accepted answers		
Generator	Min. m	Cov.	Prec.	Mean	p95	Max	Mean	Max	Wrong
iter2	1	75.7	99.4	0.97	1.70	3.40	4.55	16	0.12
	2	68.7	99.4	0.77	1.49	2.77	3.64	13	0.09
	3	56.8	99.6	0.65	1.28	2.55	3.05	12	0.07
F0	1	76.6	99.4	1.05	1.95	2.81	4.86	13	0.11
	2	68.4	99.4	0.80	1.52	2.16	3.71	10	0.08
	3	57.1	99.6	0.70	1.30	2.16	3.23	10	0.06
ForwardPref	1	75.4	98.6	0.99	1.71	2.78	4.60	13	0.09
	2	68.1	99.4	0.80	1.50	2.36	3.74	11	0.07
	3	59.7	99.3	0.70	1.28	2.14	3.27	10	0.07
STaR	1	77.4	99.2	0.97	1.72	2.80	4.51	13	0.07
	2	68.1	99.4	0.78	1.51	2.37	3.62	11	0.06
	3	59.3	100.0	0.69	1.29	2.16	3.20	10	0.05
SelfDistill	1	73.1	98.5	0.99	1.71	2.56	4.64	12	0.12
	2	67.5	99.4	0.81	1.50	2.35	3.80	11	0.10
	3	59.0	99.6	0.69	1.28	2.14	3.25	10	0.08

On a multiple-choice pool, re-paired recovered answers still match the proposed letter often, at 23.4% coverage against 41.4% unshuffled, since a letter can match by chance.

G.2 RESAMPLING IN THE SAME DIRECTION INSTEAD

Table 21: Same-direction resampling on the identical pools and rows. Each control re-solves from the question alone with three samples capped at 1536 tokens; iter2, long uses a 4096-token cap.

	Rows		Accepted			Rate (%)		
Generator	All	Correct	All	Correct	Wrong	Base acc.	Cov.	Prec.
Full pool, three samples								
iter2	470	453	128	127	1	96.4	27.2	99.2
iter2, long	470	453	268	267	1	96.4	57.0	99.6
F0	462	441	129	129	0	95.5	27.9	100.0
ForwardPref	467	448	118	118	0	95.9	25.3	100.0
STaR	464	444	121	120	1	95.7	26.1	99.2
SelfDistill	468	446	119	119	0	95.3	25.4	100.0
Band subsets of 200 rows								
F0	199	179	111	108	3	89.9	55.8	97.3
ForwardPref	189	172	113	111	2	91.0	59.8	98.2
STaR	189	169	111	106	5	89.4	58.7	95.5
SelfDistill	192	171	106	102	4	89.1	55.2	96.2

Table 21's iter2 control uses the same 470 rows as the backward probe. At one agreement, the union of their accept sets is 357 rows against the backward rule's 356, and the one row only the same-direction rule accepts is correct.

G.3 Which part of the probe prompt carries the decision

Table 22: Prompt-component ablation of the backward probe on a 100-row probe. The last four columns count rows the full prompt accepts that a control does not, and rows a control accepts that the full prompt does not, against each of the two controls.

	Rows accepted			Full against answer only		Full against trace only	
Threshold	Full	Answer only	Trace only	Full only	Control only	Full only	Control only
$m \geq 2$	77	44	46	33	0	32	1
$m \geq 3$	64	36	30	29	1	35	1

Table 22 has no accepted wrong answers in any of its four comparisons, in the full prompt, in either control and in the union, and a control adds at most one accepted row that the full prompt does not. Both prompt components contribute acceptances that the other component alone does not recover.

H Answer formatting and unreadable draws

Tables 23 and 24 quantify missing extractions and their effect on agreement rates. Excluding unreadable draws conditions these rates on successful extraction.

H.1 Draws with no extractable answer, per bank

Table 23: Share of draws with no extractable answer per vote bank.

Bank	Model	Problems	Draws	Empty rate
	GSM8K			
Boxing	iter2, 8B	1319	10552	0.065
No instruction	iter2, 8B	1319	10552	0.411
Boxing	Qwen3-1.7B	1319	10552	0.002
	Minerva			
First half	iter2, 8B	136	1088	0.063
Second half	iter2, 8B	136	1088	0.044
All problems	Qwen3-1.7B	272	2176	0.011
	MATH-500			
First half	iter2, 8B	246	1968	0.095
First half	Qwen3-1.7B	246	1968	0.012
	Second model lineage			
No instruction	Llama-3.1-8B-Instruct	500	4000	0.680
Boxing, 4k cap	Llama-3.1-8B-Instruct	500	4000	0.444
Boxing, 8k cap	Llama-3.1-8B-Instruct	375	3000	0.447

Table 23 shows that adding the boxing instruction from Appendix A.4 reduces empty draws from 41.1% to 6.5% for the fine-tuned 8B model on GSM8K. The stock 1.7B model with that instruction has 0.2% empty draws on the same problems.

H.2 What the convention for unreadable draws costs

Table 24 compares the deployed convention, which counts an empty draw as not agreeing, with an alternative that thresholds over valid draws only, and that conditions on extraction success.

Table 24: Completeness and leak in percent on one 1319-row bank, under both conventions and on both the bank drawn without the boxing instruction and the bank drawn with it.

	Empty counts as not agreeing		Empty excluded	
Bank	Completeness	Leak	Completeness	Leak
No instruction	58.3	17.8	98.5	31.9
Boxing	94.0	23.3	98.3	31.0

Table 24 shows that, within the instructed bank, excluding empty draws raises completeness from 94.0% to 98.3% and leak from 23.3% to 31.0%. Missing extractions also differ by class: without the instruction, 40.8% of correct-answer draws and 44.0% of wrong-answer draws are empty; with it, the shares are 4.4% and 24.7%. Thus the extraction convention changes both rates used to price an action.

I SECOND BENCHMARKS AND SECOND DOMAINS

I.1 DATASETS

The second mathematics benchmark is `math-ai/minervamath`, split `test`, with 272 problems (Lewkowycz et al., 2022). Probe signals are joined by index and vote banks by the full problem text. The easier mathematics benchmark is `openai/gsm8k`, configuration `main`, split `test`, with 1319 problems, 136 wrong answers and a base accuracy of 89.7% (Cobbe et al., 2021). Probe signals exist for a 500-row subset, which the price fit uses. The code domain uses HumanEval, at 375 candidates, and MBPP, at 2490 candidates (Chen et al., 2021; Austin et al., 2021). We score three fresh regenerations per candidate by label-free input and output equivalence against the candidate program, and we use the hidden tests as labels for analysis only.

J LIMITATIONS

Our evaluation uses stored answers to mathematics and program-synthesis problems with checkable labels. The binomial certificate requires independent calibration units and decision rules fixed independently of them; our experiments with multiple answers per problem report nominal tests and realised held-out risk. Prices also depend on the answer population and cost model. Applying PriceCheck to further model families, tasks or deployment settings therefore calls for refitting the prices and recalibrating on representative data (Tibshirani et al., 2019).