ON THE THEORY OF LLM-ASSISTED PROBLEM SOLVING

Anonymous authors
Paper under double-blind review

ABSTRACT

What is the most effective way to use an LLM to solve a math problem? If the problem is easy, then "one-shotting," i.e. simply posing the query, is likely to be enough. However, if the problem is harder, empirical evidence suggests that such a naive approach has a low probability of success. Instead, researchers use multiple agents and harnesses, to break the problem into subcomponents, explore different approaches, verify progress, and then combine these into a final solution.

In this work we give a formal framework for exploring such harnesses. We provably show that the optimal approach for solving a problem depends on its inherent difficulty as well as the power of the LLM-solver, and that harnesses are helpful when the problem is sufficiently harder than what an LLM can solve directly. Perhaps surprisingly, our analysis identifies a key property governing the power of such explorations: it is not merely the model's ability to solve mathematical statements, but its ability to reliably estimate their difficulty.

1 INTRODUCTION

How do LLMs solve mathematical problems? While there has been some success in simply asking them and "one-shotting" the solution, much of the progress has been achieved by orchestrating multiple queries together in a precise order. In this work we propose a formal model for such harnesses and orchestrations, defining the different roles that agents can play. The central question is how to combine all of these in a (provably) optimal way. In the process, we also develop insights for what information is critical for good orchestrations.

At its core, an LLM is a probabilistic, universal oracle. One can ask any question and receive some answer. The oracle is not without its flaws — the answer may not always be correct, the result is often non-deterministic, and a proof may not be found within the available computation budget. The question then, is how to harness the oracle's power to solve ever more difficult problems.

From a practical perspective, there has been a lot of focus in guiding the LLMs solving the problems. These *harnesses* aim to coordinate multiple agents with diverse capabilities working on the same problem to improve the overall capabilities of the system.

Some examples of such approaches include the FUNSEARCH framework (Romera-Paredes et al., 2024), which, rather than asking a model to produce mathematical proofs or full code in a single prompt, utilizes an island-based evolutionary algorithm. In geometry, ALPHAGEOMETRY (Trinh et al., 2024) couples a transformer language model to an explicit, non-neural symbolic engine to successfully solve International Mathematical Olympiad (IMO) problems, while ALPHAPROOF (AlphaProof and AlphaGeometry teams, 2024) leverages the Lean interactive theorem prover (de Moura & Ullrich, 2021) with reinforcement learning to reach IMO silver-medal standards. Beyond these closed-loop systems, the broader community has achieved milestone breakthroughs in competitive mathematics and automated deduction. For instance, DEEPSEEK-MATH (Shao et al., 2024) pushes the boundaries of open-source mathematical reasoning through self-consistent reinforcement learning, and NUMINAMATH (Beeching et al., 2024) demonstrates state-of-the-art olympiad-level problem solving via tool-integrated reasoning. In formal interactive theorem proving, LEANDOJO (Yang et al., 2023) provides an open-source playground for retrieval-augmented language models to prove theorems and interact with formal verifiers. More recently, STELLAR COLOSSEUM (Lin et al., 2026) implements an inference harness for long-horizon conjectures, integrating readiness gating, section-level subproblem DAGs, and targeted adversarial falsifiers. Rather than attempting a monolithic

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.

proof, this harness directs multi-agent teams to challenge intermediate lemmas. Outside of the math domain, structured harnesses also translate theoretical reasoning into laboratory execution: COSCIENTIST (Boiko et al., 2023) automates complex multi-step chemical reactions without human intervention, and THE AI SCIENTIST (Lu et al., 2024) represents a fully automated framework for end-to-end scientific discovery.

The use of harnesses to improve system capabilities is not surprising. Pólya's classical account of mathematical problem solving (Pólya, 1957) advocates breaking the problem down into simpler versions, verifying the partial solutions, and then collating them to obtain the full result. Evidence shows that AI-assisted theorem proving proceeds in a similar manner. The chances of "one-shotting" a hard problem are remarkably low; to stand a chance, the AI agents must first decompose the problem into easier questions, and then prove and verify individual steps.

The question we ask in this work is what are the key components in sequencing these computations? Given some partial progress on a problem, what quantities drive the decision about which path should be explored next?

We identify several building blocks. The generic LLM oracle takes as input a question/query, and context, and (probabilistically) produces an output. We focus on three types of agents, widely used in practical harnesses such as *Stellar Colosseum* (Lin et al., 2026), that can together be used to solve a problem:

- A *prover* attempts to one-shot solve a problem. This is a probabilistic algorithm, whose success probability is fixed for each action in our model.
- A *verifier* takes in a question and a proof and checks that the output is correct. While the verifier is also non-deterministic, in this work we assume that it has access to a Lean or other formal verification system, and thus succeeds with probability 1.
- A *decomposer* takes a question, along with the context, and produces a set of additional sub-questions, that, when solved together, imply the original statement.

In the stochastic setting, we additionally study the value of *cost guidance*: estimates of the cost of direct proof and of completing a proposed decomposition. We show that the ability of the model to estimate these costs is crucial for effective orchestration.

The central difficulty is orchestration: making all of the different agents work effectively together. For a hard theorem, the probability that a prover finds a complete proof in one shot may be exponentially small. Verification prevents the system from building on invalid proof nodes. A decomposer may help by exposing useful intermediate statements, but a proposed decomposition can be wrong, irrelevant, or too weak to simplify the proof.

As an initial theoretical study, we deliberately consider minimal models that already expose nontrivial orchestration tradeoffs. Proof attempts have fixed, independent success probabilities, verification is sound and free, and newly obtained proofs are not shared between distinct subproblem occurrences. We first study a fixed decomposition tree, and then allow repeated decomposition calls to reveal alternative proposals. These abstractions isolate the costs of executing and discovering proof plans. Context-dependent capabilities, summarization, and reusable intermediate lemmas are natural and very interesting future directions for extending the framework.

Our Contribution. In this work, we provide a formal theoretical framework for LLM-assisted problem solving, modeling the harness as an online orchestration algorithm interacting with probabilistic prover, verifier, and decomposer oracles.

We identify two settings. In the first, the decomposition of a hard statement into a series of easier statements is deterministic, and so the key question is when to stop the decomposition. In the second, this decomposition is stochastic, and so an additional option available to the system is to stop the current proof approach and obtain a new decomposition.

In the *deterministic decomposition setting*, we show that the offline optimum always admits an optimal static proof plan. For instances of full height at most H, we introduce an online algorithm—*ancestor dovetailing*—that systematically interleaves direct proof attempts across ancestral paths with descendant exploration. We prove that this algorithm achieves an optimal competitive ratio of $H + 1$, matching an information-theoretic lower bound for all online algorithms without cost estimates, even

if the entire decomposition tree is revealed for free. In the absence of a height bound, we show that ancestor dovetailing achieves an expected cost of $O(\mathrm{OPT}^2)$, and that this quadratic dependence is asymptotically tight.

In the *stochastic decomposition setting*, where queries sample alternative subproblem proposals from an unknown distribution, we establish that structural height bounds no longer suffice: an unguided online algorithm can be forced to incur an exponential cost even when the oracle cost is linear. To circumvent this impossibility, we formulate the trade-off between proposal exploration and proof execution via optimal stopping and Pandora's box problems. We characterize the optimal stopping threshold and show that a simple *value-guided* stopping policy equipped with fixed, uniformly α-approximate cost estimates achieves expected cost at most $(H+1)\alpha^{2H}\,\mathrm{OPT}$ on instances of full height at most H, where OPT is the optimal expected cost of a distribution-aware oracle (recovering a factor of $H+1$ under exact costs). We further show that this factor is tight for our algorithm. Together, our results quantify the fundamental limits of inference harnesses and formalize the critical role of evaluative feedback in multi-agent problem solving. In addition, our model and contributions open up interesting directions for future work, such as incorporating context management, summarization, and retrieval into the framework.

2 Related Work

Our work is situated at the intersection of automated theorem proving, multi-agent LLM orchestration, and search theory under uncertainty. We review the additional related work for the latter two topics below.

2.1 Multi-Agent Orchestration and Iterative Refinement

Our formalization of decomposers and provers is conceptually related to advanced prompting paradigms and autonomous research frameworks. Rather than linear reasoning chains, the Tree of Thoughts (ToT) paradigm (Yao et al., 2023) enables language models to explore multiple reasoning paths, self-evaluate progress, and perform lookahead search. Similarly, Self-Refine (Madaan et al., 2023) utilizes model-generated critique and feedback to iteratively improve initial outputs.

More broadly, our analysis of stochastic decomposers relates to fully autonomous research agents. The AI Scientist (Lu et al., 2024) automates the entire scientific research lifecycle, from idea generation and experimentation to manuscript preparation. In a similar vein, Google's Paper Assistant Tool (PAT) (Jayaram et al., 2026) deploys inference-scaled agents to evaluate manuscript rigor and identify mathematical flaws. Recent surveys and case studies (Woodruff et al., 2026; Feng et al., 2026) highlight that the main bottleneck in autonomous science is the exponential explosion of possible research paths during proposal generation. Our theorem on stochastic decomposition without side information (Theorem 5.1) provides a mathematical explanation for this bottleneck, proving that unguided search over alternative decompositions requires exponential cost, and showing how cost estimates can scale down this complexity.

2.2 Pandora's Box, Search Theory, and Optimal Stopping

On the theoretical front, our work draws heavily from and contributes to search theory under uncertainty. The challenge of choosing whether to continue exploring alternative subproblem proposals or to invest effort into proving an already-revealed set of assumptions is mathematically equivalent to the repeated, cost-minimizing version of the classical **Pandora's Box problem**, first solved by Weitzman (1979). This framework has been extensively generalized in computer science to understand the "price of information" in combinatorial optimization (Singla, 2018). We adapt Weitzman's reservation value rule to the recursive tree setting of theorem proving. Our "recursive value-guided proof search" algorithm extends these classic search policies to hierarchical decomposition graphs, proving that even with approximate, noisy cost estimates, we can achieve strong competitive guarantees relative to the optimal offline oracle.

3 PRELIMINARIES AND PROBLEM FORMULATION

Let $\mathcal{S}$ denote the set of mathematical statements, and let $r \in \mathcal{S}$ be the target statement to be proved. The algorithm maintains a set of *certified statements* that have been formally verified. Statements initially provided as true (axioms or known lemmas) are leaves in the proof search and have supplied certificates available at no cost. Initially, only r and these facts are revealed; operations are available only at revealed statements.

Faced with an uncertified statement $a \in \mathcal{S}$, an algorithm may invoke the LLM agents through three primitive operations:

1. **Direct proof attempt:** Succeeds with probability $q_a \in [0, 1]$. Upon success, a is immediately certified.
2. **Decomposition call:** Returns and reveals a decomposition into a set of sub-statements $A \subset \mathcal{S}$ (we will refer to these as *assumptions*). The call produces no certificate that the decomposition is correct (i.e. sound and complete).
3. **Conditional proof attempt:** For $A \neq \emptyset$, the action (a, A) is available only after assumptions A exist for statement a. It attempts to prove the implication $(\bigwedge_{b \in A} b) \implies a$, succeeding with probability $q_{a,A} \in [0, 1]$. The resulting certificate is retained while the premises are proved. Once every premise $b \in A$ is certified, a becomes certified.

When $A = \emptyset$, conditional proof is identical to direct proof, so we identify $q_{a,\emptyset} = q_a$. Every generated certificate is checked by a deterministic, sound verifier (e.g., Lean) at zero cost. Proof attempts are independent, with fixed success probabilities. Each LLM call (decomposition or proof attempt) incurs a unit cost. For an online algorithm ALG, we denote by C_{ALG} its random total cost to certify r; expectations include proof outcomes, decomposition samples, and the algorithm's randomness. We use $1/0 = \infty$.

We distinguish between two structural models of decomposition: *deterministic* (where a statement has a unique, fixed decomposition) and *stochastic* (where repeated queries yield independent proposals from a distribution).

3.1 DETERMINISTIC DECOMPOSERS

In the deterministic regime, a decomposer returns a fixed assumption set $D(a) = A_a$ for each uncertified statement a. An unproved terminal statement satisfies $D(a) = \emptyset$ and still requires a direct proof. The complete set of available decompositions forms a fixed, finite directed tree rooted at r, where initially certified facts serve as terminal leaves.

Full Height. We define the *full height* H of an instance to be the maximum number of nodes that are not initially certified on any root-to-leaf path in the complete decomposition tree. This is fixed by the input, including branches the algorithm never uses. A target r that is initially certified has $H = 0$; an unproved terminal node has $H = 1$; and a statement whose children are all initially certified also has $H = 1$.

The Offline Benchmark (OPT). The online algorithm knows neither the probabilities $q_a, q_{a,A}$ nor the decomposition tree in advance. We benchmark it against an *offline oracle* that knows the complete tree and all success probabilities, but not the outcomes of future coin flips. It starts with the same certified facts, pays for the same operations, and obeys the same revelation rules. Let $V(a)$ be its minimum expected cost at a fresh node a, and write $\mathrm{OPT} = V(r) < \infty$.

A *static proof plan* T uses supplied certificates at initially certified leaves. At every other node it chooses direct proof, or decomposition followed by a conditional proof and recursive proofs of all assumptions. In Proposition 4.1, we show that the oracle has an optimal static plan:

$$\mathrm{OPT} = \min_T C(T), \quad \text{where} \quad C(T) = |I(T)| + \sum_{v \in N(T)} \frac{1}{q_T(v)}. \tag{3.1}$$

Here, $N(T)$ denotes the nodes utilized by T that are not initially certified, $I(T) \subseteq N(T)$ is the subset of nodes decomposed by T, and $q_T(v)$ is the success probability of the prover action chosen at v (direct proof if $v \notin I(T)$, or conditional proof if $v \in I(T)$).

Example 3.1. To make the above concrete, consider the following example. Let r assert that $f(x) = x + 1$ on $\mathbb{Z}$ is bijective. A deterministic decomposer returns $D(r) = \{b, c\}$, where b is injectivity and c is surjectivity. Certifying r via decomposition requires: (i) one decomposition call, (ii) proving b, (iii) proving c, and (iv) obtaining the conditional certificate $b \wedge c \implies r$. If all proof attempts succeed with probability 1 and neither b nor c is initially certified, the plan has cost 4. The oracle compares this against a direct proof attempt of bijectivity at expected cost $1/q_r$. Figure 1 illustrates this route.

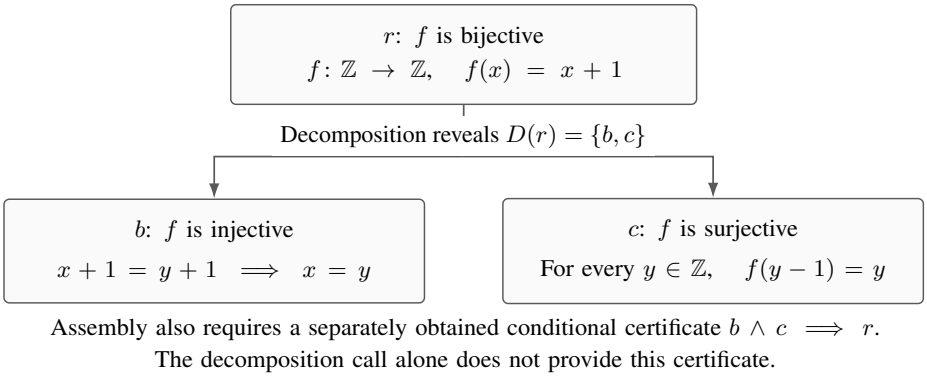


Figure 1: A proof tree for bijectivity of $f(x) = x + 1$. The arrows show the assumptions revealed by decomposition. Proving both children and obtaining the conditional certificate certifies the root.

3.2 Stochastic Decomposers

In practical systems, prompt variations, temperature sampling, and model non-determinism cause decomposers to suggest different decompositions on repeated queries. Formally, querying a stochastic decomposer at any uncertified statement a draws an independent sample $A \sim Q_a$, where Q_a is a fixed distribution over finite assumption sets. Every call costs one unit, including repeated or empty outputs. An unproved terminal statement has $Q_a(\emptyset) = 1$. The algorithm is not told which statements are terminal, and one empty output does not imply that future outputs are empty.

Exploration vs. Execution. Stochastic decomposers introduce a fundamental dilemma absent in the deterministic case: an algorithm must decide whether to invest effort into proving assumptions from an already observed proposal A, or to pay for additional calls to sample an alternative (potentially simpler) proposal $A' \sim Q_a$.

The Stochastic Benchmark (OPT)**.** The online algorithm knows neither the distributions Q_a nor the proof probabilities. The oracle knows both, but not future samples or proof outcomes. Both may retain decompositions, interleave work on them, and resume sampling, subject to the same costs, supplied facts, revelation rules, and prohibition on sharing new proofs. Unlike the deterministic case, the oracle's choice of proof route may depend on the decompositions it samples; accordingly, the benchmark optimizes over adaptive policies rather than fixed proof trees. As before, $V(a)$ is the oracle's minimum expected cost at a fresh node and $\mathrm{OPT} = V(r)$. In Theorem 5.2, we show that an optimal oracle can sample first and then complete one chosen route at each node.

Example 3.2 (The cost of finding a useful decomposition)**.** Suppose a decomposition has expected completion cost of 2 with probability $1/4$ and 14 otherwise, including the conditional proof and all its assumptions. If these costs are known on arrival, waiting for cost 2 takes four sampling calls in expectation and has total expected cost $4 + 2 = 6$. A direct proof that costs 5 in expectation is cheaper, even though $5 > 2$: finding a useful decomposition also costs work.

Completion Costs For the positive stochastic results, we assume there is no infinite chain of nested subproblems and every sampled decomposition has finite expected completion cost almost surely. Initially certified facts have $V(a) = 0$. At any other node, we define

$$X_a(A) = \frac{1}{q_{a,A}} + \sum_{b \in A} V(b), \qquad d_a = \frac{1}{q_a}. \tag{3.2}$$

Here $X_a(A)$ is the expected cost of completing decomposition A: its conditional proof and its separate child obligations. It excludes the call that returned A. The value d_a is the expected direct-proof cost, and $X_a(\emptyset) = d_a$. The oracle knows these costs; the guided algorithm in Section 5.3 receives estimates of them.

4 DETERMINISTIC DECOMPOSERS

We first study the deterministic model and benchmark defined in Section 3.1. Even in this setting, the algorithm faces a basic choice: it can keep trying to prove the statement directly, or it can spend time proving a collection of simpler statements. A failed proof attempt does not reveal whether direct proof is merely difficult or impossible. Our main result shows that it is enough to try both possibilities at every level of the decomposition tree. The resulting competitive ratio is exactly $H + 1$, where H is the full height measured by the number of statements not initially certified along a path. This factor is necessary even if the entire tree is revealed for free. Without a height bound, the same algorithm has expected cost $O(\mathrm{OPT}^2)$, and this quadratic dependence is also tight.

4.1 THE OPTIMAL PROOF PLAN

Before designing our online algorithm, it is useful to characterize its offline optimum. An oracle endowed with full knowledge of the decomposition tree and all success probabilities could, in principle, execute an adaptive policy that dynamically switches actions based on intermediate trial failures. The following result shows that this adaptivity is unnecessary: the oracle's optimal strategy always reduces to a fixed, static proof plan. This allows us to characterize OPT via a clean dynamic programming recurrence that weighs the cost of a direct proof attempt against the combined cost of decomposing and recursively certifying its premises.

Proposition 4.1 (The optimal proof plan)**.** *The deterministic benchmark has an optimal static proof plan. Its value satisfies $V(a) = 0$ for initially certified facts, while*

$$V(a) = \min\left\{\frac{1}{q_a},\ 1 + \frac{1}{q_{a,A_a}} + \sum_{b \in A_a} V(b)\right\} \tag{4.1}$$

for every other statement. At an unproved terminal statement the sum is zero and $q_{a,\emptyset} = q_a$, so $V(a) = 1/q_a$. Consequently, $\mathrm{OPT} = V(r) = \min_T C(T)$*. Optimal plans need not be unique, but they all have the same expected cost* OPT.

The proof is a straightforward backwards induction argument, which we present in Appendix A.

4.2 AN OPTIMAL ALGORITHM FOR BOUNDED HEIGHT

We first give the proof for the case when the height, H, of the decomposition is known. The algorithm interleaves a direct proof at every level of the discovered proof tree with making progress on a proof by decomposition. At a high level this resembles a rent-or-buy decision: continue paying for direct attempts, or invest in another way to finish the proof, though in this case the cost of the decomposition route is itself unknown.

Ancestor dovetailing

Initialize. Start with the target r and all initially supplied certificates.
Repeat until r is certified:

(1) Starting at r, perform an in-order traversal of the uncovered proof tree, stopping at the first uncertified node, which we call the *frontier*.

(2) Try one direct proof at each node on root to frontier path. At the first success, retain the proof, propagate certification, and end the round.

(3) If all direct attempts failed and the frontier has a revealed empty decomposition, end the round. Otherwise call the decomposer on the frontier node, or make a conditional proof attempt if the decomposition already exists.

The algorithm requires neither the height nor the proof probabilities. We write C_{AD} for its random total cost, namely the number of operations it performs until the target is certified.

Theorem 4.2 (Tight height bound)**.** *For every instance of full height at most H, ancestor dovetailing has expected cost at most*

$$(H+1)\,\mathrm{OPT}\,.$$

For every integer $H \geq 1$, no randomized online algorithm has a competitive ratio smaller than $H+1$ on this class.

The proof follows by charging each set of up to H direct attempts to the corresponding optimal action, and we detail it in Appendix B.1.

4.3 Without a height bound

The same algorithm gives a guarantee when the decomposition tree is arbitrarily tall. In this case, the relevant parameter is the expected work of the oracle itself.

Corollary 4.3. *Let $B = \mathrm{OPT}$. Ancestor dovetailing has expected cost $O(B^2)$, without knowing B or having a height bound. This dependence is tight: for every sufficiently large scale b and every randomized online algorithm, there is an instance with $\mathrm{OPT} = \Theta(b)$ on which the algorithm has expected cost $\Omega(b^2)$.*

We defer the proof to Appendix B.2

5 Stochastic decomposers

We now study the stochastic model in Section 3.2, where each call samples a decomposition from a distribution. This models an LLM decomposer whose suggestions vary across calls. We show that a critical piece is an estimate of the cost of the remaining proof. (Intuitively, this is similar to human experts making a decision on which proof path is likely to succeed.) Without such guidance, finding a useful proof path may require exponentially many attempts. Estimates of the remaining proof costs allow substantially better guarantees: with exact estimates, our recursive algorithm is $(H+1)$-competitive, independent of the number of possible decompositions. For approximate estimates, the bound depends only on H and the estimates' accuracy. We emphasize that the algorithm need not know either parameter.

5.1 Why side information is necessary

Even if every call returns at most one assumption, a stochastic decomposer can offer exponentially many possible proof paths. The next theorem shows why generating decompositions is not enough: the algorithm also needs guidance about which ones are inexpensive to complete.

Theorem 5.1 (Decomposition without side information)**.** *For all integers $m \geq 2$ and $h \geq 1$, there is a family of stochastic decomposition instances of full height $h+1$ such that, for every randomized online algorithm, there exists an instance satisfying*

$$\mathrm{OPT} \leq h(m+1)+1, \qquad \mathbb{E}[C_{\mathrm{ALG}}] = \Omega(m^h).$$

The competitive ratio is therefore $\Omega(m^{h-1}/h)$.

The proof follows a 'needle in a haystack' idea, where the algorithm needs to exhaustively search all possible paths, whereas the optimal approach can be much more direct. We give the formal argument in Appendix C.1.

5.2 The optimal policy and optimal stopping

From a proof plan to an adaptive policy. In the deterministic setting, randomness only determines how many attempts the chosen prover actions need: failures leave the state unchanged, and the oracle knows all success probabilities. An optimal proof tree can therefore be fixed before execution. With stochastic decompositions, the available proof routes depend on the outcomes of paid decomposition

calls. Even an oracle that knows the distributions may need to adapt its choice to the decompositions it observes. It could also interleave sampling with partial proof work, retain several unfinished routes, and resume sampling after making progress. Thus OPT optimizes over policies that depend on the observed history, rather than fixed proof trees.

Our main structural result, Theorem 5.2, shows that this larger policy space nevertheless admits a simple optimal policy: at each statement, the oracle either proves it directly or samples decompositions until a reservation criterion is met, then completes the selected route recursively. The result extends the classical Pandora reservation rule to recursive proof search by showing that interleaving partial execution with further sampling cannot improve the expected cost under our independence and no-sharing assumptions. This commitment is local to each recursive call; the oracle need not sample the entire proof tree before doing proof work. The policy also guides our algorithm in Section 5.3.

We now proceed to describe this in more detail. A returned decomposition A has expected completion cost $X_a(A)$ from (3.2). We first solve the choice between completing an available route and paying for another decomposition. We then show that this choice characterizes the optimal proof policy.

An important observation is that this problem is closely related to Pandora's box and the price of information. Specifically, this is the repeated, identically distributed version of Pandora's box: sampling reveals the cost of an option, and previously sampled options remain available. We use its classical reservation rule in the cost-minimization form.

Allow a general sampling cost $c > 0$; decomposition calls in our model have $c = 1$. A sample reveals an independent copy of a finite nonnegative completion cost X. A direct-proof alternative has expected cost d, with $d = \infty$ if it is unavailable. The *reservation value* ρ is the unique solution of

$$\mathbb{E}[(\rho - X)^+] = c, \qquad (z)^+ = \max\{z, 0\}. \tag{5.1}$$

The left side is continuous, starts at zero, and grows without bound; it is strictly increasing wherever it is positive. The equation balances the sampling cost with the expected saving over a route of cost ρ.

This argument leads directly to the following Theorem, we give the full proof in Appendix C.2.

Theorem 5.2 (A sample-then-commit optimal policy)**.** *Under the independent, no-sharing stochastic model of Section 3.2, assume that there is no infinite chain of nested subproblems and every sampled decomposition has finite expected completion cost almost surely. Initially certified nodes have $V(a) = 0$. At any other node a, let $d_a = 1/q_a$ and let ρ_a solve*

$$\mathbb{E}_{A\sim Q_a}[(\rho_a - X_a(A))^+] = 1$$

if decomposition is available; otherwise set $\rho_a = \infty$. Then

$$V(a) = \min\{d_a, \rho_a\}.$$

At a terminal node, $X_a(\emptyset) = d_a$ and $\rho_a = d_a + 1$, so direct proof is optimal. An optimal oracle uses the following rule at each node not initially certified:

- *If $d_a \le \rho_a$, attempt direct proof until success, without making any decomposition calls.*
- *If $d_a > \rho_a$, sample decompositions until the first A with $X_a(A) \le \rho_a$. Obtain its conditional certificate and prove each child by applying this rule recursively.*

When Q_a is deterministic, $X_a(A)$ is constant and the reservation equation gives $\rho_a = 1 + X_a(A)$, recovering the deterministic recurrence (4.1).

5.3 Recursive value-guided decomposition

We now give a competitive algorithm that takes advantage of the approximate cost estimates in guiding its search. Our algorithm follows the structure of the optimal policy in Theorem 5.2: choose a route, then complete it recursively.

Before going into details, we make precise the additional information available to the algorithm. The algorithm is not given the proof probabilities $q_a, q_{a,A}$ or the distributions Q_a. Instead, when a is revealed, it is given an estimate $\widehat{d}_a$ of the direct-proof cost, and an estimate $\widehat{X}_a(A)$ when a

decomposition A is returned. For the guarantee, assume that there exists some (unknown)$\alpha \geq 1$ that bounds the estimates' errors:

$$\frac{d_a}{\alpha} \leq \widehat{d}_a \leq \alpha d_a, \qquad \frac{X_a(A)}{\alpha} \leq \widehat{X}_a(A) \leq \alpha X_a(A). \tag{5.2}$$

The estimates reveal no future proof outcomes or decomposition samples. Each estimate is fixed for its node and route, so a repeated decomposition receives the same score. An empty output uses the direct-proof estimate: $\widehat{X}_a(\emptyset) = \widehat{d}_a$.

Adaptive Algorithm The oracle's Pandora reservation threshold approach required the knowledge of the decomposition distribution and exact completion costs. We adapt it to use *cost estimates, whose accuracy α is unknown*. The high level idea is that after t unit-cost decomposition calls, we select the cheapest estimated route whose estimate is at most $t + 1$; otherwise we sample again. This stopping rule uses neither the distribution nor accuracy α.

Recursive value-guided proof search: $\mathrm{Prove}(a)$

(1) **Handle initial facts.** If a is initially certified, return its supplied certificate.

(2) **Initialize the options.** Set $t = 0$. Option 0 is direct proof, with score $\widehat{d}_a$.

(3) **Sample or select.** Let $\widehat{M}_t$ be the minimum score among options $0, \ldots, t$. If $\widehat{M}_t \leq t + 1$, select a minimum-score option, breaking ties by smallest index, and go to step (4). Otherwise, pay one unit to sample a decomposition, increment t, store its output A_t, and score $\widehat{X}_a(A_t)$ as option t. Repeat step (3). Count and store repeated and empty outputs too.

(4) **Complete the selected route.** Stop sampling at a. For option 0 or an empty decomposition, repeat direct attempts until success. Otherwise, obtain the selected conditional certificate by repeated attempts, then call $\mathrm{Prove}(b)$ for each selected child node b, sequentially in a fixed order. Return the proof of a once the selected route is complete.

Note that the counter t counts only decomposition calls at the current node. The number made before selecting a route is

$$T = \inf\{t \geq 0 : \widehat{M}_t \leq t + 1\}. \tag{5.3}$$

We let C_{VG} denote the random variable of the total number of operations performed by the entire recursive algorithm.

Theorem 5.3 (Recursive value guidance)**.** *On an instance of full height H the recursive value-guided proof search has expected total cost*

$$\mathbb{E}[C_{\mathrm{VG}}] \leq (H + 1)\alpha^{2H}\,\mathrm{OPT}\,.$$

In particular, exact estimates give $(H + 1)\,\mathrm{OPT}$.

We defer the full proof to Appendix C.3, and show that with exact estimates, the factor $H + 1$ is tight for this algorithm.

6 Conclusion

Our results separate two sources of difficulty in proof orchestration: allocating effort within a fixed decomposition structure, and discovering which of several proposed approaches is worth pursuing. In the deterministic setting, ancestor dovetailing achieves tight guarantees without knowing proof difficulties. In the stochastic setting, a distribution-aware optimum admits a local sample-then-commit policy, while cost guidance supports competitive recursive search. Important next questions include whether more effective guided algorithms can avoid the height dependence, and how the picture changes with costly or imperfect evaluation and context-dependent agents.

AI Declaration

The model and main algorithmic ideas in this paper were developed by the human authors. AI tools assisted in developing the algorithmic details, constructing the proofs, and writing the paper. The authors edited and revised the initial text proposed by the AI tools.

References

AlphaProof and AlphaGeometry teams. AI achieves silver-medal standard solving International Mathematical Olympiad problems. https://deepmind.google/blog/ai-solves-imo-problems-at-silver-medal-level/, 2024. Accessed: 2026-09-25.

Edward Beeching, Shengyi Costa Huang, Albert Jiang, Jia Li, Benjamin Lipkin, Zihan Qina, Kashif Rasul, Ziju Shen, Roman Soletskyi, and Lewis Tunstall. Numinamath 7b tir. https://huggingface.co/AI-MO/NuminaMath-7B-TIR, 2024.

Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023. doi: 10.1038/s41586-023-06792-0.

Leonardo de Moura and Sebastian Ullrich. The Lean 4 theorem prover and programming language. In *Automated Deduction – CADE 28*, volume 12699 of *Lecture Notes in Computer Science*, pp. 625–635. Springer, 2021. doi: 10.1007/978-3-030-79876-5_37.

Tony Feng, Trieu H. Trinh, Garrett Bingham, Dawsen Hwang, Yuri Chervonyi, Junehyuk Jung, Joonkyung Lee, Carlo Pagano, Sang hyun Kim, Federico Pasqualotto, Sergei Gukov, Jonathan N. Lee, Junsu Kim, Kaiying Hou, Golnaz Ghiasi, Yi Tay, YaGuang Li, Chenkai Kuang, Yuan Liu, Hanzhao Lin, Evan Zheran Liu, Nigamaa Nayakanti, Xiaomeng Yang, Heng-Tze Cheng, Demis Hassabis, Koray Kavukcuoglu, Quoc V. Le, and Thang Luong. Towards autonomous mathematics research. *arXiv preprint arXiv:2602.10177*, 2026. doi: 10.48550/arXiv.2602.10177.

Rajesh Jayaram, Drew Tyler, David Woodruff, Corinna Cortes, Yossi Matias, Vahab Mirrokni, and Vincent Cohen-Addad. Towards automating scientific review with Google's Paper Assistant Tool. *arXiv preprint arXiv:2606.28277*, 2026. doi: 10.48550/arXiv.2606.28277.

Honghao Lin, David P. Woodruff, Yuan Deng, Jieming Mao, Song Zuo, and Vahab Mirrokni. Stellar Colosseum: A many-agent harness for long-horizon research in mathematics and theoretical computer science. *arXiv preprint arXiv:2609.15983*, 2026. doi: 10.48550/arXiv.2609.15983.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024. doi: 10.48550/arXiv.2408.06292.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594, 2023. doi: 10.52202/075280-2019.

George Pólya. *How to Solve It: A New Aspect of Mathematical Method*. Princeton University Press, Princeton, NJ, second edition, 1957.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. doi: 10.48550/arXiv.2402.03300.

Sahil Singla. The price of information in combinatorial optimization. In *Proceedings of the Twenty-Ninth Annual ACM–SIAM Symposium on Discrete Algorithms (SODA)*, pp. 2523–2532, 2018. doi: 10.1137/1.9781611975031.161.

Trieu H. Trinh, Yuhuai Wu, Quoc V. Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024. doi: 10.1038/s41586-023-06747-5.

Martin L. Weitzman. Optimal search for the best alternative. *Econometrica*, 47(3):641–654, 1979. doi: 10.2307/1910412.

David P. Woodruff, Vincent Cohen-Addad, Lalit Jain, Jieming Mao, Song Zuo, MohammadHossein Bateni, Simina Branzei, Michael P. Brenner, Lin Chen, Ying Feng, Lance Fortnow, Gang Fu, Ziyi Guan, Zahra Hadizadeh, Mohammad T. Hajiaghayi, Mahdi JafariRaviz, Adel Javanmard, Karthik C. S., Ken ichi Kawarabayashi, Ravi Kumar, Silvio Lattanzi, Euiwoong Lee, Yi Li, Ioannis Panageas, Dimitris Paparas, Benjamin Przybocki, Bernardo Subercaseaux, Ola Svensson, Shayan Taherijam, Xuan Wu, Eylon Yogev, Morteza Zadimoghaddam, Samson Zhou, Yossi Matias, James Manyika, and Vahab Mirrokni. Accelerating scientific research with Gemini: Case studies and common techniques. *arXiv preprint arXiv:2602.03837*, 2026. doi: 10.48550/arXiv.2602.03837.

Kaiyu Yang, Aidan M. Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. LeanDojo: Theorem proving with retrieval-augmented language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 21573–21612, 2023. doi: 10.52202/075280-0944.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 11809–11822, 2023. doi: 10.52202/075280-0517.

APPENDIX

A PROOF OF PROPOSITION 4.1

Proof. A state records the revealed decompositions, conditional certificates, and certified nodes. There are finitely many states, and a useful successful operation only adds information or certificates. A failed attempt leaves the state unchanged. Because the oracle knows the success probabilities, there is an optimal deterministic stationary policy: it fixes one action for each state and repeats that action after a failure. Indeed, one can construct this policy backward from completed states, choosing the action that minimizes its expected waiting time until success plus the optimal cost of its successor state.

Under this policy, the sequence of successful operations is fixed; randomness only determines how many failed attempts precede each success. Extract a certificate for r from this sequence. Stop at an initially certified fact and use its supplied proof. At a node certified conditionally, retain its decomposition call, conditional proof, and certificates for every child. At a node certified directly, retain its direct proof. Discard operations unused by the extracted certificate. The resulting static plan has no larger expected cost, and every static plan is itself a feasible policy. Thus an optimal static plan exists.

Finally, an initially certified fact has value zero. At every other node, a plan either proves a directly at expected cost $1/q_a$, or pays for its decomposition, conditional proof, and the children whose proofs are not supplied. Minimizing between these choices gives (4.1); for an empty decomposition the second expression is dominated by direct proof. □

B PROOFS OF THE DETERMINISTIC DECOMPOSITION

B.1 PROOF OF THEOREM 4.2

Proof of the upper bound. Each round either advances an optimal static plan or certifies one of its subtrees by another route. Since a round costs at most H direct attempts plus one frontier operation, it suffices to bound the number of rounds by the number of operations in an execution of that plan. The extra frontier operation is where the $+1$ comes from.

Fix an optimal static plan $T^\star$. For each of its selected prover actions, independently sample the number G_v of trials until its first success. Thus G_v is geometric on $\{1, 2, \dots\}$ with mean $1/q_{T^\star}(v)$. Couple the algorithm to these same trial streams whenever it uses a selected action. The realized execution length of the plan is

$$Z = |I(T^\star)| + \sum_{v \in N(T^\star)} G_v, \qquad \mathbb{E}[Z] = \mathrm{OPT}\,. \tag{B.1}$$

These Z operations comprise all decomposition calls and all trials through the first success of each selected action. Whenever the algorithm executes one of them, mark it completed. If it certifies a plan node by another route, discard the remaining planned operations in that subtree.

Each round completes or discards at least one remaining planned operation. To see this, consider its active path. If the path contains a direct leaf of $T^\star$, the round attempts its selected direct action unless an earlier direct attempt succeeds. If it contains no direct leaf, every node on the path is decomposed by the plan, so the frontier operation belongs to the plan unless a direct attempt succeeds first.

The only remaining case is an earlier direct success at a node u that the plan decomposes. This certifies u and discards its remaining planned operations. At least one such operation must remain: otherwise the plan's certificates, together with any children already certified by other routes, would certify u, contradicting its presence on the unresolved path. Thus an interrupted round also eliminates at least one remaining planned operation. No operation is completed or discarded twice.

The number of rounds is consequently at most Z. Each round costs at most H direct attempts and one frontier operation. Hence the cost is at most $(H + 1)Z$ for every coupled realization, and taking expectations proves the bound. □

Proof of the lower bound. Consider a chain of H initially unproved statements $a_0, \dots, a_{H-1}$, with target a_0, followed by an initially certified fact z:

$$D(a_i) = \{a_{i+1}\} \quad (0 \le i < H-1), \qquad D(a_{H-1}) = \{z\}.$$

The tree has $H+1$ nodes, but its height is H because z is initially certified. All conditional actions above a_{H-1} succeed with probability one. There are $k = H+1$ remaining candidate actions: the direct proof at each a_i, and the final conditional proof of a_{H-1} from $\{z\}$. The latter uses z as an explicit premise, so it is distinct from direct proof and has its own success probability. Choose one of these actions uniformly at random, give it success probability $\delta \in (0,1)$, and give every other candidate probability zero. The oracle knows which action can succeed. At most H decomposition calls and $H-1$ probability-one conditional proofs expose that action and connect its proof to the root. The certificate for z costs nothing. The oracle then needs $1/\delta$ trials in expectation, so its expected cost is at most $2H + 1/\delta$.

For the online algorithm, grant every decomposition and every probability-one conditional certificate for free. This only makes its task easier. It must still find a success among k actions, exactly one of which succeeds with probability δ. Fix its internal randomness and consider the transcript in which every candidate attempt fails. Let $n_i(t)$ count attempts at candidate i among the first t operations of this transcript. If τ is the time of the first candidate success, then

$$\Pr(\tau > t) = \frac{1}{k}\sum_{i=1}^{k}(1-\delta)^{n_i(t)} \ge (1-\delta)^{t/k}.$$

The inequality follows from convexity and $\sum_i n_i(t) \le t$. It remains valid after averaging over the algorithm's randomness. Therefore

$$\mathbb{E}[\tau] \ge \sum_{t\ge 0}(1-\delta)^{t/k} = \frac{1}{1-(1-\delta)^{1/k}} \sim \frac{k}{\delta} \qquad \text{as } \delta \longrightarrow 0.$$

Some member of the distribution has at least this expected online cost, while every member has oracle cost at most $2H + 1/\delta$. Letting δ tend to zero rules out every ratio below $H+1$. □

B.2 Proof of Corollary ??

Proof. Retain the coupling and random variable Z from (B.1). The number of rounds is still at most Z. Before round t, at most $t-1$ decompositions have been revealed, so the active path has at most t nodes. Thus

$$C_{\mathrm{AD}} \le \sum_{t=1}^{Z}(t+1) = \frac{Z^2+3Z}{2}.$$

The plan $T^\star$ is fixed, so its number of decomposition calls $|I(T^\star)|$ is constant and contributes no variance to Z. The randomness comes only from the waiting times G_v for its selected proof actions. Write $q_v = q_{T^\star}(v) > 0$; these probabilities are positive because the plan has finite expected cost. A geometric waiting time counting all attempts up to and including the first success has variance $\mathrm{Var}(G_v) = (1-q_v)/q_v^2$. The proof-attempt streams are independent, so the variances add:

$$\begin{aligned}\mathrm{Var}(Z) &= \sum_{v\in N(T^\star)} \mathrm{Var}(G_v) = \sum_{v\in N(T^\star)} \frac{1-q_v}{q_v^2} \\ &\le \sum_{v\in N(T^\star)} \frac{1}{q_v^2} \le \left(\sum_{v\in N(T^\star)} \frac{1}{q_v}\right)^2 \le B^2.\end{aligned}$$

The first inequality uses $1-q_v \le 1$. The second holds because expanding the square adds only nonnegative cross terms. For the last inequality, the sum of the expected waiting times is $\sum_v 1/q_v = B - |I(T^\star)| \le B$. Finally, the identity $\mathbb{E}[Z^2] = \mathrm{Var}(Z) + (\mathbb{E}[Z])^2$, together with $\mathbb{E}[Z] = B$, gives $\mathbb{E}[Z^2] \le 2B^2$ and hence $\mathbb{E}[C_{\mathrm{AD}}] \le B^2 + 3B/2$. If the target is not initially certified, every proof plan contains at least one proof attempt, so $B \ge 1$, proving the upper bound. An initially certified target has $B = 0$ and is returned at zero cost.

For the lower bound, fix an integer $m \geq 2$ and use the chain above with $H = m$. Consider the family of $m + 1$ instances obtained by varying which candidate action can succeed: one of the m direct proofs, or the final conditional proof of a_{m-1} from the certified fact z. In each instance, the chosen candidate has success probability $1/m$, all other candidates have probability zero, and all conditional actions above a_{m-1} have probability one. Thus the instances share the same decomposition tree and differ only in the location of the candidate that can succeed.

Every instance has oracle cost between m and $3m$: any proof requires a success from the chosen candidate, taking m attempts in expectation, and the oracle needs at most $2m$ additional operations to expose that action and connect its proof to the root. Choose an instance uniformly from these $m + 1$ possibilities. The preceding survival bound, with $k = m + 1$ and $\delta = 1/m$, gives

$$\mathbb{E}[\tau] \geq \frac{1}{1 - (1 - 1/m)^{1/(m+1)}} \geq \frac{m+1}{-\log(1 - 1/m)} \geq m^2 - 1.$$

The last two inequalities use $1 - e^{-x} \leq x$ for $x \geq 0$ and $-\log(1-1/m) \leq 1/(m-1)$. Consequently, the family has expected online cost at least $m^2 - 1$, while its oracle cost is $\Theta(m)$. Taking $m = \lceil b \rceil$ proves the claimed $\Omega(b^2)$ lower bound for an instance with $\mathrm{OPT} = \Theta(b)$. □

C Proofs for Stochastic Decomposers

C.1 Proof of Theorem 5.1

Proof. Consider a complete m-ary tree with h edges on each root-to-leaf path and $n = m^h$ leaves. At an internal node, a decomposition call returns one uniformly random child. The children are alternative decompositions, not assumptions that must all be proved. Every conditional proof succeeds with probability one.

There is one instance for each choice of a distinguished leaf ℓ^*. Its direct proof succeeds with probability one; every other direct proof has probability zero. At a leaf, every decomposition call returns the empty set. No statement is initially certified, so the full height is $h + 1$.

The oracle knows ℓ^* and follows the path to it. At each level, it needs m decomposition calls in expectation to obtain the correct child, followed by one conditional proof. Proving ℓ^* costs one more operation, giving the bound $h(m + 1) + 1$.

For the online lower bound, choose ℓ^* uniformly among the n leaves. Give the algorithm the entire tree and every conditional certificate for free. It must still find ℓ^* by direct proof attempts. Fix its random choices and consider the first $t = \lfloor n/2 \rfloor$ attempts on the transcript in which they all fail. This list contains at most t distinct leaves and is independent of ℓ^*. Hence

$$\Pr[C_{\mathrm{ALG}} > t] \geq 1 - \frac{t}{n} \geq \frac{1}{2}, \qquad \mathbb{E}[C_{\mathrm{ALG}}] \geq t \Pr[C_{\mathrm{ALG}} > t] \geq \frac{n}{8}.$$

The expectation averages over the chosen instance and the algorithm's randomness, so at least one instance has the stated online cost. □

C.2 Proof of Theorem 5.2

We begin by recalling the classical results on Pandora's box.

Lemma C.1 (Classical Pandora reservation rule Weitzman (1979); Singla (2018))**.** *The minimum expected sampling and completion cost is* $\min\{d, \rho\}$*. If* $d \leq \rho$*, choose direct proof. Otherwise, sample until the first value* $X \leq \rho$ *and complete that decomposition. Writing* $p = \Pr[X \leq \rho]$*, its expected cost is*

$$\frac{c}{p} + \mathbb{E}[X \mid X \leq \rho] = \rho. \tag{C.1}$$

Proof. The threshold rule takes $1/p$ samples in expectation, then pays completion cost $\mathbb{E}[X \mid X \leq \rho]$. The reservation equation rewrites as $c = \rho p - \mathbb{E}[X \mathbf{1}_{\{X \leq \rho\}}]$, giving (C.1).

For optimality, let b be the least cost among direct proof and all sampled options, initially $b = d$, and track $P = \min\{b, \rho\}$. A new independent sample decreases P by $(P - X)^+$, whose conditional

expectation is at most $\mathbb{E}[(\rho - X)^+] = c$. For any policy with finite expected cost, let N be its number of samples. Summing these nonnegative decreases gives

$$\min\{d, \rho\} \le c\mathbb{E}[N] + \mathbb{E}[P_N] \le \mathbb{E}[\text{total cost}],$$

since the selected option has conditional expected cost at least P_N. The potential is bounded by ρ, so the argument also applies when $d = \infty$ or X is unbounded. Policies of infinite expected cost already satisfy the bound. Direct proof or the threshold rule attains it. □

We are now ready for the proof of the main Theorem.

Proof. We prove the claim from children to parents. Initially certified nodes cost zero. At a terminal node, decomposition calls only return the empty set and add cost before the same direct proof, so $V(a) = d_a$. Now fix a nonterminal node a and assume the claim holds for its descendants. Put $v_a = \min\{d_a, \rho_a\}$. By the induction hypothesis, completing a returned decomposition using the stated child policies costs $X_a(A)$ in expectation. The Pandora reservation rule in Lemma C.1 therefore gives a policy of cost v_a, proving $V(a) \le v_a$.

The other inequality needs a separate argument. The induction hypothesis describes the optimum for a fresh child, but a competing policy at a could work on that child only partially, abandon its route, and try another decomposition. To cover all such policies, we track a remaining-work quantity that starts at v_a, reaches zero when a is proved, and decreases by at most one in expectation per paid operation.

At each revealed node b, set $U_b = 0$ if it is certified. Otherwise set

$$U_b = \min\left\{ d_b, \rho_b, \min_{\substack{\text{retained } A \\ A \neq \emptyset}} \left(s_{b,A} + \sum_{u \in A} U_u \right) \right\},$$

where $s_{b,A}$ is zero if the conditional certificate has been obtained and $1/q_{b,A}$ otherwise. The three terms represent direct proof, further sampling, and completion of a retained decomposition. A minimum over no retained decompositions is infinity. Initially $U_a = v_a$; each newly revealed child u is untouched and has $U_u = V(u)$ by induction.

- A direct attempt reduces U_b to zero on success. Its expected decrease is $q_b U_b \le 1$, since $U_b \le d_b = 1/q_b$; if $q_b = 0$, there is no decrease.
- A conditional attempt reduces its route's remaining work by at most $1/q_{b,A}$ on success, which occurs with probability $q_{b,A}$. Taking the minimum over routes cannot increase this reduction, so the expected decrease of U_b is at most one.
- A new decomposition has untouched children, so its remaining cost is $X_b(A)$. It reduces U_b by $(U_b - X_b(A))^+ \le (\rho_b - X_b(A))^+$. The reservation equation bounds the expected decrease by one. Empty or repeated outputs change nothing.

Suppose an operation at b decreases U_b by Δ. At its parent, the cost of the route containing b drops by Δ, while all other route costs stay unchanged: the no-sharing rule makes b a child of just one route. The minimum defining the parent's value can therefore drop by at most Δ. Repeating this argument up to a shows that U_a also drops by at most Δ. Thus each paid operation decreases U_a by at most one in expectation. To prove a, a policy must reduce this value from v_a to zero.

To add these decreases over the run, let C be the cost of any policy with finite expected cost. Let D_t be the decrease in U_a at its t-th operation, with $D_t = 0$ if it has already finished. An operation occurs only when $C \ge t$, and its conditional expected decrease is at most one, so $\mathbb{E}[D_t] \le \Pr[C \ge t]$. The decreases are nonnegative and sum to v_a, since the policy eventually proves a. Consequently,

$$v_a = \sum_{t \ge 1} \mathbb{E}[D_t] \le \sum_{t \ge 1} \Pr[C \ge t] = \mathbb{E}[C].$$

The final sum is the expected number of paid operations. Policies of infinite expected cost already satisfy the bound. Hence $V(a) = v_a$, and the stated policy is optimal. □

C.3 Proof of Theorem 5.3

Proof. We separate the cost of choosing a route from the cost of completing it. First we bound the sampling work and the optimal completion cost of the route selected at one node. We then use induction on height to account for the algorithm's own recursive work.

Choosing a route. Fix a node a that is not initially certified. Let T be its number of decomposition calls and Z the optimal completion cost of the selected route: d_a for direct proof, or $X_a(A)$ for a decomposition A. In particular, Z uses optimal costs for the children. We will show

$$\mathbb{E}[T] \le \alpha V(a), \qquad \mathbb{E}[Z] \le \alpha^2 V(a). \tag{C.2}$$

This argument applies to terminal nodes as well (because the algorithm does not know if a node is terminal): they may incur sampling costs, and all their outputs are empty.

Write $\widehat{Z} = \widehat{M}_T$ for the estimate of the selected route. The accuracy assumption and the stopping rule give, respectively,

$$Z \le \alpha \widehat{Z}, \qquad \widehat{Z} \le T + 1.$$

First compare with direct proof, assuming $d_a < \infty$. The direct option is always available with estimate $\widehat{d}_a$. If $T > 0$, the algorithm did not stop just before its last sample, so $T < \widehat{M}_{T-1} \le \widehat{d}_a$. The bound $T < \widehat{d}_a$ also holds when $T = 0$. Since the algorithm selects a minimum-score option, $\widehat{Z} \le \widehat{d}_a$. Therefore

$$T \le \alpha d_a, \qquad Z \le \alpha \widehat{Z} \le \alpha \widehat{d}_a \le \alpha^2 d_a.$$

The first factor of α converts the selected estimate into an upper bound on its true cost; the second accounts for the error in the direct-proof estimate.

Next compare with the oracle's sampling policy. Use the same sequence of decomposition samples for both policies, continuing it hypothetically if our algorithm stops first. Let N be the first sample whose true completion cost is at most ρ_a, and let Y be that cost. The reservation rule gives

$$\mathbb{E}[N + Y] = \rho_a. \tag{C.3}$$

Every route requires at least one proof attempt, so $Y \ge 1$. If $T \le N$, then $T+1 \le N+1 \le N+\alpha Y$. If $T > N$, the oracle's selected route is already available at time $T-1$, with estimate at most αY. Since our algorithm has not stopped at that time, $T < \widehat{M}_{T-1} \le \alpha Y$. As $N \ge 1$, both cases imply

$$T + 1 \le N + \alpha Y. \tag{C.4}$$

Taking expectations and using $\alpha \ge 1$ gives

$$\mathbb{E}[T] \le \alpha \rho_a, \qquad \mathbb{E}[Z] \le \alpha \mathbb{E}[T+1] \le \alpha^2 \rho_a.$$

Together with the direct-proof comparison and $V(a) = \min\{d_a, \rho_a\}$, this proves (C.2).

Completing the route. Let R_h bound the algorithm's expected cost relative to $V(a)$ at nodes of height at most h. Set $R_0 = 1$, since height-zero nodes are initially certified and cost nothing. Suppose the bound holds at height $h-1$. Conditional on the chosen route, the recursive calls use fresh independent proof and decomposition streams, so the induction hypothesis applies to each child. For a selected decomposition A, its expected execution cost is at most

$$\frac{1}{q_{a,A}} + R_{h-1} \sum_{b \in A} V(b) \le R_{h-1} X_a(A),$$

where the last inequality uses $R_{h-1} \ge 1$. Direct proof costs d_a, so in either case the expected execution cost is at most $R_{h-1} Z$. Adding the sampling cost and applying (C.2) gives

$$\mathbb{E}[T] + R_{h-1}\mathbb{E}[Z] \le \big(\alpha + \alpha^2 R_{h-1}\big) V(a).$$

Thus we may take $R_h = \alpha + \alpha^2 R_{h-1}$. This recurrence gives $R_h = h + 1$ for exact estimates and $R_h \le (h+1)\alpha^{2h}$ in general. Applying the bound at the target, where $V(r) = \mathrm{OPT}$, proves the theorem.

□

Height dependence of this algorithm. While it is open whether any guided algorithm can improve the height dependence, we show that our analysis of this particular algorithm is tight with perfect estimates. Consider a chain $a_1 \to \cdots \to a_H \to f$, where f is initially certified and every decomposition deterministically returns the next node. Direct proofs are impossible. Conditional proofs cost one in expectation except at a_H, where the expected cost is an integer $M \geq 2$. At a node with h uncertified nodes remaining, the sole decomposition has exact score $X_h = M + 2h - 2$. Its sampling counter starts at zero, so it makes $X_h - 1 = M + 2h - 3$ decomposition calls before selecting this route, even though every call returns the same output. The counter resets at the next child: the calls spent at its parent do not count toward its own stopping threshold. Thus the total number of decomposition calls is

$$\sum_{h=1}^{H} (M + 2h - 3) = HM + H(H - 2).$$

The conditional proofs cost another $M + H - 1$ in expectation, giving

$$\mathbb{E}[C_{\mathrm{VG}}] = (H + 1)M + H(H - 1) - 1, \qquad \mathrm{OPT} = M + 2H - 1.$$

The bottom proof's cost M appears in every ancestor's completion estimate, causing roughly M sampling calls at each of the H levels; executing that proof costs another M. The oracle instead makes just one decomposition call per level. For fixed H, the ratio tends to $H + 1$ as $M \to \infty$.