

KNOWING WHEN TO STOP: IGP-BENCH AND THE MULTI-AGENT SYSTEM GALOIS FOR THE INVERSE GALOIS PROBLEM

Shaowu Zhang* Jan Safrata Brent Kong Eric Lu
 Sathvik Redrouthu Caiman Moreno-Earle Tony Yue Yu†
 California Institute of Technology

ABSTRACT

Frontier agents can now sustain hours of autonomous work. Greater capability may bring greater progress, but it may also carry an agent farther along a wrong route. Existing long-horizon mathematics benchmarks are difficult to build, saturate quickly, or rely on language models to grade partial progress, making it difficult to measure how much effort an agent spends on a wrong route. We introduce **IGP-Bench**, a benchmark of 200 long-horizon inverse Galois problems from computational algebraic geometry, 39 of which have published explicit polynomials or regular families in our surveyed literature. Each problem has intermediate results that are difficult to find but easy to check with deterministic GAP and Magma verifiers. Agents record their partial results and attempts in an artifact graph, allowing us to measure verified progress, instruction following, tool use, and resource allocation and, most importantly, to analyze their ability to **know when to stop**. Across five problems, 27 verification checkpoints, and more than 32,000 possible routes, we observe that knowing when to stop and construction abilities are largely distinct (Spearman $\rho = +0.01$) and that newer models can spend substantial effort on wrong routes. To address this failure mode, we introduce **GALOIS, an open-source multi-agent system** designed to identify promising routes and stop work on wrong routes. It comprises a lead agent, an optional strategy reviewer, specialist agents equipped with 36 general algebraic geometry skills, and a route-rejection agent that checks for possible obstructions in parallel. In reference runs led by Claude Opus 5.5, GALOIS obtains Magma-verified realizations for all five tasks in 10–126 minutes, including polynomials for M_{23} and M_{24} that no single agent realizes in our evaluation. We will release the tasks, verifiers, evaluation code, and GALOIS.

1 INTRODUCTION

AI agents can now work autonomously for hours on tasks of increasing difficulty and duration (Kwa et al., 2025), but greater capability does not necessarily bring greater awareness of when they are doing things wrong. For example, OpenAI reported that a long-running model mistakenly sustained an intrusion into Hugging Face’s infrastructure for two and a half days, illustrating how persistence on a wrong route could be dangerous (OpenAI, 2026b; Hugging Face, 2026; OpenAI, 2026a; Greenblatt et al., 2026). Similar issues have been observed in AI-assisted mathematical research. In the recent realization of M_{23} as a Galois group over $\mathbb{Q}$ by Huang et al. (2026), the most capable models were also the most persistent on routes that eventually failed. According to the authors’ AI disclosure and reflection (Zhang, 2026; Chiou, 2026), the agents kept proposing to go further on a wrong route, and it was the human authors who decided when to stop computations on wrong routes and identified the essential obstruction, which led to the final success. This raises a question: do AI agents know when to stop pursuing a wrong route, and how well can they make this judgment?

*Corresponding author: szhang7@caltech.edu

†Principal investigator

Inverse Galois theory provides a particularly suitable setting for evaluating whether research agents know when to stop. These problems have recently become a focus of AI-assisted mathematical research: SAIR’s IGP24 competition drew 256 participants over two months to search for explicit degree-24 polynomials realizing transitive groups of degree 24 over $\mathbb{Q}$ (SAIR Foundation, 2026); AIM’s 2026 workshop on AI and number theory posed the question of realizing the last sporadic simple group M_{23} (American Institute of Mathematics, 2026); Epoch AI features the M_{23} problem in FrontierMath: Open Problems (Epoch AI, 2026); and *Scientific American* also recently covered the development of this field (Chiou, 2026). As experts in inverse Galois theory point out (Antieau, 2026), an essential difficulty is deciding when to stop a computation and move to another candidate.

To introduce IGP-Bench, we first recall some mathematical background. For a finite group G , the inverse Galois problem asks whether there is a polynomial $f \in \mathbb{Q}[V]$ whose Galois group is isomorphic to G . We call an approach to constructing such a polynomial a *route*. Each route is specified by a list of conjugacy classes of G , called a *class vector*, for example $(2A, 23A, 23B)$ of M_{23} . Our literature survey provides a reference set of known routes, while historical attempts suggest that a generic route is unlikely to succeed. The difficulty is that a class vector may pass the initial group-theoretic tests yet lead to a cover that cannot be defined over $\mathbb{Q}$. Simply choosing a class vector and pushing its construction further therefore need not bring the target polynomial within reach. This makes route selection and stopping decisions a substantive part of the mathematics. The case of M_{23} illustrates the challenge (Figure 2): its 16 nontrivial conjugacy classes give 816 three-point and 3,876 four-point routes, while only $(2A, 23A, 23B)$ is currently known to yield a realization over $\mathbb{Q}$ (Huang et al., 2026). An agent must therefore decide which candidates justify further computation and which should be set aside as new evidence emerges.

IGP-Bench uses these problems to evaluate agents. Each of its 200 tasks asks for a realization of a finite group over $\mathbb{Q}$. We divide the work into three stages: *screening*, which identifies candidate routes and tests their group data; *route identification*, which verifies the group data for a route labeled correct; and *construction*, which follows a selected route through its intermediate objects toward a polynomial. A task comes with a *checkpoint ladder*: a sequence of intermediate objects checked by computer-algebra verifiers. For each problem, agents record intermediate results and attempted routes in a labeled *artifact graph*, with resource costs such as CPU hours and output tokens attached to artifacts.

From these artifacts, we compute a progress score P and assess instruction following and tool use. We quantify stopping behavior with the knowing-when-to-stop metric K_r , and resource use with the resource efficiency metric U_r , without an LLM judge (Appendix D).

We evaluated nine frontier-model systems on IGP-Bench, spanning open-weight and proprietary models: Claude Opus 5, Claude Fable 5.1, Claude Sonnet 5, GPT-6 Sol, GPT-6 Astra, GPT-5.6 Sol, GPT-5.6 Terra, DeepSeek V4 Pro, and GLM-5.3 Flash. We ran each system twice on each of five tasks targeting the Mathieu groups M_{11} , M_{12} , M_{22} , M_{23} , and M_{24} , for 90 single-agent runs in total. Each run had a 10-hour wall-clock limit, a 500,000-output-token cap, and access to eight CPUs. Mean progress scores ranged from 2.6 to 81.0, with Claude Opus 5 at the top (81.0), outperforming the newer Claude Fable 5.1 (62.4), GPT-6 Sol (62.1), and GPT-6 Astra (59.4) (Figure 1). Among many other findings in Section 6, the following two are particularly interesting.

1. **Going deeper does not always imply a better score.**
2. **Choosing a route and knowing when to stop are each largely distinct from going deeper.**

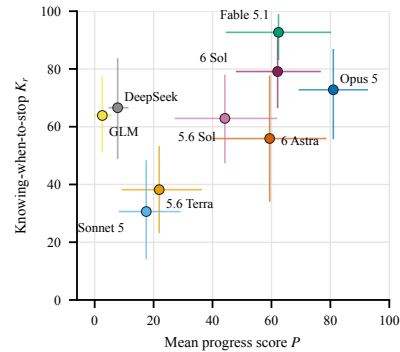


Figure 1: Progress and knowing when to stop by system, means with 95% bootstrap intervals.

We also introduce GALOIS, an open-source multi-agent system. Its lead agent plans the work, while specialist agents follow stage-specific procedures based on general methods from the literature. In parallel, a route-rejection agent checks candidate routes for known obstructions. GALOIS rejects a route only when an exact obstruction is proved; it may also defer work when a resource limit is reached.

In separate reference runs led by Claude Opus 5.5, GALOIS obtained verified realizations for all five tasks within 126 minutes each, including polynomials for M_{23} and M_{24} that no single agent realized in our evaluation. Our contributions are IGP-Bench and its metrics (Section 3), GALOIS (Section 4), and the initial evaluation (Sections 5–7).

2 RELATED WORK

Mathematics benchmarks and partial credit. FrontierMath and Riemann-Bench score final answers without partial credit, and HorizonMath ranks solutions that improve a published bound (Glazer et al., 2024; Garre et al., 2026; Wang et al., 2026). IMProofBench pairs research-level proof problems, graded by human experts, with automatically graded follow-up subquestions, and TheoremBench evaluates machine-verified theorems in Lean (Schmitt et al., 2025; Pham et al., 2026). PaperBench and EXP-Bench grade parts of research workflows with LLM judges (Starace et al., 2025; Kon et al., 2026). In IGP-Bench, the intermediate states are the mathematical objects on a task’s checkpoint ladder, and GAP or Magma verifies them (Table 1). The M_{23} realization is prior work (Huang et al., 2026; Zhang, 2026).

AI systems, research agents, and multi-agent systems. FunSearch and AlphaEvolve find new constructions by search (Romera-Paredes et al., 2024; Novikov et al., 2025). QED, an open-source multi-agent system, separates planning, proving, and verification and has produced original proofs on research-level projects (An et al., 2026); Aletheia and the AI co-mathematician also target research-level mathematics (Feng et al., 2026; Zheng et al., 2026). MetaGPT, AutoGen, and CAMEL assign agents different roles (Hong et al., 2024; Wu et al., 2023; Li et al., 2023). GALOIS also has a lead agent and specialists, but it differs in two respects: every mathematical result must pass verification, and a route is rejected only when an exact obstruction is proved, while resource limits may trigger deferral; a dedicated agent checks routes for obstructions in parallel. Appendix A lists further work.

Cost, overthinking, and stopping. Kapoor et al. (2025) argue that accuracy-only benchmarks can reward systems that are unnecessarily expensive. Reasoning models can spend computation where it is not needed (Chen et al., 2025b; Sui et al., 2025), and extended reasoning in agentic tasks can displace useful interaction with the environment (Cuadron et al., 2025). Adapting test-time computation can work better than allocating it uniformly (Snell et al., 2024), while longer reasoning can reinforce distractions and mistaken strategies (Gema et al., 2025). The abstention benchmark AgentAbstain scores one decision per task: on paired everyday tasks, the agent should either act or abstain, and each task is scored pass or fail (Liu et al., 2026). IGP-Bench measures the outcome of many decisions per run: over hours, the agent repeatedly decides whether to continue a route, and K_r summarizes what its wrong routes cost, with routes assessed using literature support or a verified polynomial produced during the run (Table 1).

Table 1: Related benchmarks. *Long-horizon*: an agent works on one task for hours (Δ : multi-step runs that are shorter or of unreported length). *Partial credit*: intermediate progress earns credit. *Deterministic*: grading uses neither human nor LLM judges. *Task-derived*: the intermediate states are objects produced on the way to a solution, not subquestions written separately. *Stopping*: whether and how stopping decisions are measured. $\checkmark$ yes, Δ partly, $\times$ no; N/A: no partial credit is given.

Benchmark	Long-horizon	Partial credit	Deterministic	Task-derived	Stopping
FrontierMath	$\times$	$\times$	$\checkmark$	N/A	$\times$
Riemann-Bench	Δ	$\times$	$\checkmark$	N/A	$\times$
HorizonMath	$\times$	$\times$	Δ	N/A	$\times$
IMProofBench	Δ	$\checkmark$	Δ	$\times$	$\times$
TheoremBench	$\times$	$\checkmark$	$\checkmark$	$\times$	$\times$
PaperBench	$\checkmark$	$\checkmark$	$\times$	$\times$	$\times$
EXP-Bench	Δ	$\checkmark$	Δ	$\times$	$\times$
AgentAbstain	$\times$	$\times$	Δ	N/A	binary per task: act or abstain
IGP-Bench	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	continuous per route: time, tokens, CPU hours

3 IGP-BENCH

3.1 BACKGROUND

A *cover* is a finite map from a curve to the projective line. A class vector with r entries prescribes the ramification of a cover with monodromy group G branched over r points; we call the route *three-point* or *four-point* accordingly. Such a cover, when defined over $\mathbb{Q}$, gives a polynomial family $F(T, V)$ over $\mathbb{Q}(T)$. The family is a *regular* realization of G if its Galois group over $\mathbb{Q}(T)$ is G and $\mathbb{Q}$ is algebraically closed in its splitting field. By Hilbert’s irreducibility theorem, infinitely many rational values of T then give a polynomial over $\mathbb{Q}$ with Galois group G (Serre, 1992; Malle & Matzat, 2018). Group-theoretic tests can rule out a class vector before any equation is computed. Passing them yields no polynomial, and neither does an existence proof (Appendix B), so we score route identification and construction separately.

3.2 TASKS

Three tiers. Each of the 200 tasks specifies a finite group G and asks for a polynomial $f \in \mathbb{Q}[V]$ with $\text{Gal}(f/\mathbb{Q}) \cong G$. The 38 *Tier 1* tasks have a published polynomial or regular family. The 55 *Tier 2* tasks have an existence proof but no explicit polynomial in the cited sources. The 107 *Tier 3* tasks comprise 106 open tasks and $\text{PSL}_2(16)$, for which we retain the Tier 3 label assigned at the time of the experiment. A later literature audit identified a published explicit polynomial for this group; its Tier 3 label here records the experimental classification, not its current mathematical status. Domain experts designed the tasks and assigned the tiers from the cited literature.

Checkpoints. The generic checkpoint ladder has five checkpoints: a generating tuple, a braid orbit, a cover, a regular family, and a final polynomial. A *generating tuple* for a class vector $(C_1, \dots, C_r)$ consists of elements $g_i \in C_i$ that generate G and satisfy $g_1 \cdots g_r = 1$. Up to simultaneous conjugation, these tuples form the *Nielsen class* of the route. For four or more branch points a *braid orbit* corresponds to a connected family of covers; for three, each element is a single cover. The first two checkpoints concern this group data, the last three the construction. Six tasks have ladders adapted to known constructions (Appendix D). At each checkpoint, GAP or Magma checks the required properties of the submitted object; credit does not depend on a reference solution, and agents have access to none.

Route ledger. The *route ledger* lists the class vectors supported by the literature and any known obstructions. Only routes supported by the literature or newly proved valid during a run are labeled *correct*. For a newly proved route, the run must produce a verified polynomial (Appendix E). This designation determines the route-identification score and K_r (Section 3.4). Appendix D gives the acceptance rules, and Section 7 discusses the limits of this designation.

What the agent sees. The agent receives the target group, objective, checkpoint descriptions, point schedule, and a fixed budget. It works without network access in a persistent sandbox with a shell, Python, GAP, Magma, and an offline corpus of 17 documents, from which those that reveal the task’s realization are excluded. It submits intermediate results in a required format. A formatter checks the syntax, but the agent sees neither the route ledger nor any verifier result, so it must assess its own progress.

3.3 EXAMPLE: M_{23}

The M_{23} task illustrates the three stages. In *screening*, the agent lists candidate routes and tests their group data: there are 816 three-point and 3,876 four-point class vectors, and each verified fact about them earns screening credit. In *route identification*, the agent must submit, for the literature-supported route $(2A, 23A, 23B)$, group data that the verifier accepts; naming the route earns nothing. This route has seven covers, exactly one of which is defined over $\mathbb{Q}$ (Huang et al., 2026). In *construction*, the agent follows the selected route by computing the cover’s equations, expressing them over $\mathbb{Q}$, and obtaining and specializing a regular family (Appendix C). No single agent reached the exact cover equations on this route. Instead, several runs constructed, on the route $(2A, 4A, 23A)$, a cover over a quartic field; the branch cycle lemma rules this route out over $\mathbb{Q}$ (Figure 2).

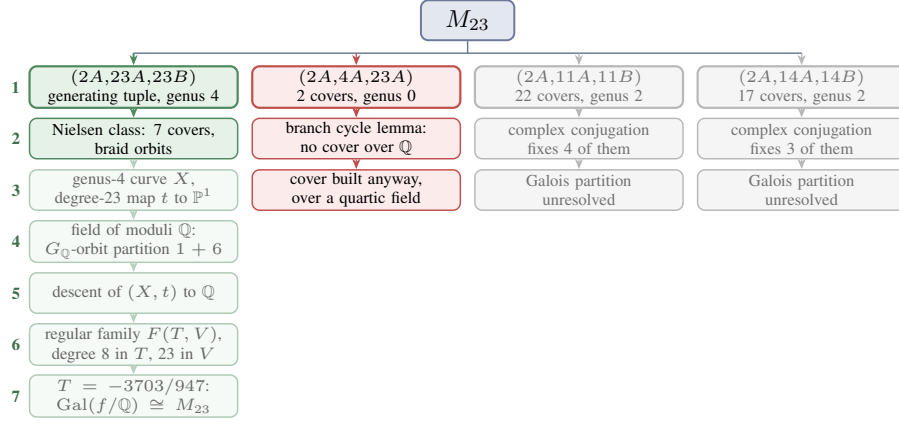


Figure 2: **Routes on M_{23} .** Green: the correct route (Huang et al., 2026), one box per checkpoint (Appendix C); full boxes were verified by at least one single-agent run, faded boxes by none, and the GALOIS reference runs complete the column. Red: a genus-0 route ruled out by the branch cycle lemma, on which several agents still built a cover over a quartic field. Grey: genus-2 routes with neither a known cover over $\mathbb{Q}$ nor a proved obstruction.

3.4 METRICS

Artifact graph. A run is one system’s attempt at one task. The agent records its work as a directed graph. Each node is an *artifact*: a declared unit of work containing tool calls, outputs, and checkpoint submissions. The agent also records which artifacts depend on which others and declares each artifact as screening or as research on a named class vector. The evaluator assigns model and tool costs to the artifact that is open and reports unattributed costs separately. Verifiers alone determine progress.

Progress score. The progress score adds three components:

$$P = S + R + C, \quad 0 \leq S \leq 30, \quad R \in \{0, 30\}, \quad 0 \leq C \leq 40. \quad (1)$$

The *screening score* S rewards verified facts about candidate routes and constructions, up to 30 points. Each fact earns credit once, with a weight based on the strength of its evidence. The *route-identification score* R gives 30 points for verified group data on a correct route, and earning R also sets S to 30. The *construction score* C depends on the deepest construction checkpoint that the verifier accepts. A verified polynomial earns all 40 construction points, regardless of how the agent found it. Appendix D gives the acceptance rules and point schedules.

Unless stated otherwise, tables report S , R , and C as percentages of their respective maxima. **POLY** is the percentage of runs that produce a verified polynomial.

Knowing-when-to-stop and resource efficiency metrics. We define knowing when to stop as ending or redirecting work on a wrong route before it uses up the budget needed for other routes. We quantify this with the *knowing-when-to-stop metric* K_r and resource use with the *resource efficiency metric* U_r . Both lie between 0 and 100; the subscript r names the resource (output tokens by default).

To compute K_r , we separate the artifacts declared as screening from those declared as research and group the latter by class vector. Let B_r be the total cost of the screening artifacts, and let $\bar{C}_r^+$ and $\bar{C}_r^-$ be the mean research costs per correct and per wrong route. Then

$$K_r = 100 \frac{B_r + \bar{C}_r^+}{B_r + \bar{C}_r^+ + \bar{C}_r^-}. \quad (2)$$

A higher K_r indicates a lower mean cost per wrong route relative to screening plus the mean cost of a correct route. Including screening allows the metric to account for useful preliminary work even

before the agent reaches a correct route. Because K_r uses the mean cost per wrong route, we report the number of researched routes with it (Appendix G.3).

U_r is the share of recorded cost spent on credited artifacts and the artifacts they depend on. Together, K_r and U_r distinguish choosing routes selectively from making verified progress on the routes chosen. Appendix D provides the accounting rules, a worked example (Figure 6), and the artifact graph of one run (Figure 7).

4 GALOIS

GALOIS builds stopping decisions into its research process in two ways: a route-rejection agent works in the background, and every experiment has a resource limit.

Control loop. At each step, the lead agent reads the plan and the evidence sheet, then proposes tool calls, specialist work, or completion. A deterministic checker validates the proposal’s format. An optional strategy reviewer checks whether the proposal is supported by the evidence, advances the work, and uses appropriate tools. The runner launches resource-limited subprocesses or specialist sessions and records their outputs in the evidence sheet (Figure 8). Each skill is a written procedure for one pipeline stage (Appendix B). The revision used for our runs contains 36 skills and 122 registered tools, covering candidate enumeration, cover reconstruction, descent, and specialization.

Rejecting and deferring. Initial screens rank candidates by generation of G , rationality, closure under complex conjugation, Nielsen count, and genus. A candidate is *rejected* only on an exact certificate, namely a field-of-moduli test, an audited obstruction to its class vector, or a verified negative result in descent or regularity, and only within the scope of that certificate; the reports of the route-rejection agent are advisory. A candidate is *deferred* when a resource limit, such as the time limit of a Gröbner call, is reached without a certificate. When exact recognition fails, the next step must be a justified precision increase, a coordinate change, another resource-limited test, or deferral, since more digits do not repair unsuitable coordinates (Appendix C, step 8); the evidence sheet records what needs repair and when to revisit it. Once all retained candidates on a route have been rejected or deferred, the scheduler moves on.

Reference runs. The ten reference runs, two per task, use a Claude Opus 5.5 lead at medium effort, with strategy review disabled and no spend cap. All ten reach a verified polynomial (Table 2). Appendix F gives the configuration and the scoring of these runs.

Table 2: **GALOIS reference runs**, two per task. *Rejected*: the three-point route excluded by an exact field-of-moduli test before the run finished the four-point route shown. Numbers are run 1 / run 2. *Steps*, *Lead*, *Obstr.*: control-loop steps, lead-agent calls, and route-rejection-agent calls; *Min*: wall-clock minutes; K_r uses output tokens over these steps.

Task	Rejected	Finished route	Steps	Lead	Obstr.	Min	USD	Tokens	K_r
M_{11}	(4A, 4A, 5A)	(2A, 2A, 3A, 4A)	46 / 45	27 / 24	10 / 10	78 / 78	26.40 / 22.33	128k / 119k	73.5 / 73.0
M_{12}	none	(3A, 3A, 6A)	13 / 13	6 / 9	6 / 5	13 / 10	6.23 / 7.27	38k / 38k	100 / 100
M_{22}	none	(2A, 11A, 11B)	22 / 25	13 / 12	5 / 8	126 / 59	24.64 / 14.51	252k / 114k	100 / 100
M_{23}	none	(2A, 23A, 23B)	16 / 17	12 / 9	3 / 4	43 / 41	7.97 / 7.27	38k / 41k	100 / 100
M_{24}	(3A, 3A, 12B)	(2A, 2A, 2A, 12B)	37 / 31	16 / 11	8 / 11	79 / 43	15.34 / 12.69	88k / 74k	35.1 / 46.7

5 EXPERIMENTAL SETUP

Tasks and systems. We evaluate on the five Mathieu tasks M_{11} , M_{12} , M_{22} , M_{23} , and M_{24} , all Tier 1. Together they have 27 checkpoints: seven for M_{23} and five for each of the other four. The August 2026 realization of M_{23} (Huang et al., 2026) was excluded from the agents’ corpus. Up to order, there are 4,752 three-point and 28,227 four-point class vectors, for 32,979 candidate routes in total. The ledger lists 13 correct routes, roughly one in 2,500. We test nine models: Claude Opus 5, GPT-6 Sol, GPT-6 Astra, Claude Fable 5.1, GPT-5.6 Sol, GPT-5.6 Terra, Claude Sonnet 5, DeepSeek V4 Pro, and GLM-5.3 Flash, all at effort *xhigh* except Fable, which runs at *max* in two later batches with the same budgets. A *system* means a model together with its harness and effort

setting. Each model uses its native Claude or Codex harness, or an OpenRouter-backed adapter. All systems access the sandbox through the same metered, offline Model Context Protocol bridge, which records every call. Each system completes two runs per task (Table 4). Appendix E describes the runs included and names the systems for which stored answers were blocked.

Sandbox, budgets, verification. Each run uses a network-isolated workspace with Python, GAP, and Magma (Appendix E). The budget per run is 10 hours, 500,000 output tokens, eight CPUs, and a task-dependent memory limit. Hitting a budget limit ends the run, but finalized artifacts can still earn credit. A checkpoint counts only when its verifier returns `verified`. Appendix E gives the verifier limits.

6 RESULTS AND ANALYSIS

IGP-Bench has the right level of difficulty. Mean progress scores range from 2.6 to 81.0 (Table 3; Figure 3). Four systems, GPT-5.6 Terra, Claude Sonnet 5, DeepSeek V4 Pro, and GLM-5.3 Flash, do some screening correctly (S between 8.7% and 49.7% of its maximum) but rarely identify a correct route (R at most 20%) and make almost no construction progress (C at most 2.5%); none of their 40 runs produces a verified polynomial. The other five systems make measurable progress on routes, each identifying a correct route in at least half of its runs, so their 50 runs can be used to compare how agents choose and pursue routes; the next four paragraphs use these runs (Appendix G.2). No system solves all five tasks, although all five are Tier 1 tasks with a published realization. Only 16 of the 90 runs produce a verified polynomial, 3 on M_{11} , 5 on M_{12} , and 8 on M_{22} . No run does so on M_{23} or M_{24} .

Table 3: **Main results on IGP-Bench.** Means over the runs of each system. S , R , and C are percentages of their maxima, 30, 30, and 40; P is in raw points. POLY is the percentage of runs ending with a verified polynomial. U_r is the resource efficiency metric and K_r the knowing-when-to-stop metric, both computed from output tokens. GALOIS[†] summarizes separate reference runs under a different protocol, and its U_r is a step-based proxy (Appendix F). Table 5 provides intervals, and Appendix D defines the other metrics.

System	Verified progress					Allocation		Protocol		Tool use		
	Screen	Route	Constr.	Total	POLY	Eff.	Stop	Format	Fidel.	Valid	Term.	Runs
	S	R	C	P	(%)	U_r	K_r	I_f	η	(%)	(%)	n
Claude Opus 5	100.0	100.0	52.5	81.0	50.0	57.7	72.8	96.9	99.0	99.8	91.2	10
GPT-6 Sol	97.0	80.0	22.5	62.1	20.0	31.4	79.1	99.2	98.4	100.0	98.7	10
GPT-6 Astra	78.0	50.0	52.5	59.4	50.0	60.3	55.9	98.2	100.0	99.8	96.7	10
Claude Fable 5.1	94.7	70.0	32.5	62.4	30.0	72.2	92.7	95.6	97.5	100.0	91.1	10
GPT-5.6 Sol	80.7	50.0	12.5	44.2	10.0	27.5	62.9	98.4	98.0	99.9	94.2	10
GPT-5.6 Terra	49.7	20.0	2.5	21.9	0.0	24.6	38.2	96.3	99.7	99.9	94.0	10
Claude Sonnet 5	45.0	10.0	2.5	17.5	0.0	26.9	30.6	93.9	97.3	99.8	95.5	10
DeepSeek V4 Pro	22.7	0.0	2.5	7.8	0.0	34.4	66.6	86.5	88.3	99.6	88.6	10
GLM-5.3 Flash	8.7	0.0	0.0	2.6	0.0	33.5	63.9	53.1	75.0	97.2	82.0	10
GALOIS [†]	100.0	100.0	100.0	100.0	100.0	82.8	82.8	N/A	N/A	N/A	N/A	10

Going deeper does not always imply a better score. GPT-6 Astra matches Claude Opus 5 on mean construction score ($C = 52.5$) and verified-polynomial rate (50%), yet ranks fourth overall with $P = 59.4$, compared with Opus’s leading score of 81.0 (Table 3); the 21.6-point gap comes entirely from screening (6.6 points) and route identification (15 points).

Choosing a route and going deeper on it are largely distinct abilities. In the progress score $P = S + R + C$, with maxima $30 + 30 + 40 = 100$, the sum $S + R$ measures how well an agent chooses a route, and the construction score C measures how far it then goes on one. Across the 50 runs of the five systems, the correlation between C and $S + R$ is only +0.08 (Spearman, 95% bootstrap CI $[-0.20, +0.35]$; Appendix G.2). For example, GPT-6 Sol and Astra have similar

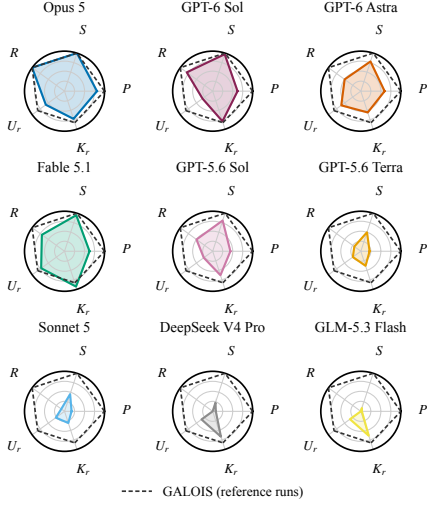


Figure 3: **Five-metric profiles of the nine systems** on a 0–100 scale. Dashed: GALOIS reference runs.

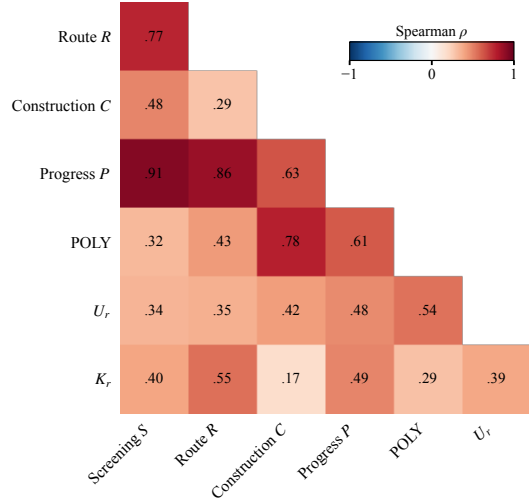


Figure 4: **Spearman correlations** over all 90 runs (K_r on 86); Figure 9 gives the full matrix.

progress scores ($P = 62.1$ versus 59.4) but very different construction scores ($C = 22.5$ versus 52.5); Sol makes up the difference mainly through route identification, identifying a correct route in eight of ten runs against Astra’s five (Table 7).

Knowing when to stop and going deeper are largely distinct abilities. Over the same 50 runs, the correlation between the knowing-when-to-stop metric K_r and the construction score C is $+0.01$ (Spearman, 95% bootstrap CI $[-0.27, +0.30]$; $+0.17$ over all runs, Figure 4). Comparing runs within the same task gives $+0.25$ ($[-0.05, +0.51]$), so going deeper explains at most a small share of K_r (Appendix G.3). Opus and Astra, tied on construction, differ in K_r (72.8 versus 55.9 ; Figure 1).

Knowing when to stop is only weakly correlated with resource efficiency. Over the same 50 runs, the correlation between the knowing-when-to-stop metric K_r and the resource efficiency metric U_r is $+0.29$ (Spearman, 95% bootstrap CI $[-0.02, +0.56]$). A high K_r alone therefore does not show that an agent converted its recorded work into verified progress.

GALOIS knows when to stop. In every reference run on M_{11} and M_{24} , GALOIS starts with a three-point route, $(4A, 4A, 5A)$ for M_{11} and $(3A, 3A, 12B)$ for M_{24} , rejects it after exact field-of-moduli tests rule out its covers, and moves to a correct four-point route, $(2A, 2A, 3A, 4A)$ and $(2A, 2A, 2A, 12B)$ respectively, which leads to a certified polynomial (Table 2). On M_{11} the rejection is cheap in computation, under one minute of tool time over 6 steps; the correct route then takes 36–37 steps. On M_{24} the rejection is the expensive part, with 12–18 steps, 53–65% of output tokens, and 52–63 minutes of tool time summed over parallel calls, after which the correct route needs 16 steps and about 15 more minutes.

Tier 2 and 3 tasks of IGP-Bench also measure agents’ ability. We run the best-performing system, Claude Opus 5, once on each of ten tasks without a correct route in the ledger, with at most 2 h 48 min of agent time per task. Five of them are Tier 2 tasks and five are Tier 3 tasks under the classification used here, which retains the experimental Tier 3 label for $\text{PSL}_2(16)$. It solves none of them, but it makes meaningful progress, even proving some candidate routes impossible, and averages $P = 45.1$ under the open-case profile (Appendix D.4; Table 11).

7 DISCUSSION AND LIMITATIONS

Mathematical ability and swarm-based research. Pursuing a mathematical conjecture reflects several abilities: choosing strategies, carrying out constructions, and knowing when to stop. When exploration is inexpensive, swarms can succeed through parallel search, but final success reveals little about individual agents’ route selection and stopping decisions. Inverse Galois problems make these decisions important: many routes compete for limited resources, and meaningful construction progress requires sustained effort, even on wrong routes. Meanwhile, group-theoretic constraints, rationality conditions, and intermediate computations provide evidence about route viability. IGP-Bench measures construction progress and how agents allocate resources as this evidence accumulates.

Route assessment and training. GALOIS separates construction from route assessment using different agents. In its M_{11} and M_{24} reference runs, the route-rejection agent disproves the initial routes and redirects work to routes that yield verified polynomials. These decisions and their mathematical justifications could serve as training examples for individual agents. Such supervision could alternatively come from a stronger model or a teacher with reference solutions or known obstructions. An open question is whether this feedback can teach agents to select routes and recognize when to stop independently.

Implications for evaluation. Benchmarks should go beyond success or failure to help explain outcomes and identify how agents’ decisions could improve. The verified intermediate results, artifact graphs, and resource records in IGP-Bench provide an example of how to examine the evidence, decisions, and resource allocation behind an outcome.

Scope and limitations. First, computational cost limits evaluation: each main single-agent run has a 10-hour budget on eight HPC cores. Limited resources prevented controlled ablations of GALOIS; its reference runs are primarily used to set benchmark parameters and evaluate agents.

Second, viable but undocumented routes may be labeled wrong and receive no route-identification credit until a verified polynomial is produced; intermediate results can still earn screening and construction credit. Domain expertise suggests that most generic candidate routes fail over $\mathbb{Q}$, and no agent established a new valid route in our experiments. These observations support the literature-based ledger as a practical reference.

Finally, the GALOIS reference runs use Claude Opus 5.5 at medium effort. Their success suggests that weaker lead models may also work, but this remains untested. Limited funding and time led us to prioritize open inverse Galois problems over testing this possibility.

8 CONCLUSION

IGP-Bench evaluates research agents through verified intermediate progress and route-level resource use. Our evaluation shows that stronger construction performance does not consistently coincide with better route selection or stopping behavior. GALOIS integrates parallel obstruction checks into a multi-agent research workflow and obtains verified realizations for all five Mathieu tasks in separate reference runs. Extending this approach to open inverse Galois problems is a direction for future work.

REPRODUCIBILITY STATEMENT

Upon publication, we will release the 200 tasks and checkpoint definitions, GAP and Magma verifiers, route ledger, scoring profiles, evaluation and metric code, agent manual, task briefs, MCP benchmark bridge, and full GALOIS source. The fixed corpus consists of cited, published documents, which we redistribute only where their licenses permit. Section 5 and Appendix E describe the software requirements, the sandbox, the budgets, the system settings, and the runs included. Appendices C and F document the M_{23} checkpoint ladder and the GALOIS run configuration. We will also publish the preserved run directories, including artifact graphs, checkpoint submissions, tool logs, and abandoned artifacts. If a system produces a verified realization of a Tier 3 group whose realization remains open, we will disclose the system, model, and configuration that produced it. Reproduction has two practical limits: Magma-based verifiers require a Magma license, and frontier-model outputs are nondeterministic and may change with provider updates. Agent trajectories can therefore be reproduced only in distribution, not exactly.

AI USE STATEMENT

We used generative AI to help draft and revise the text, write $\text{\LaTeX}$, develop scripts for processing run data and generating tables and figures, search the literature, and check the manuscript against the released data and code. We did not use it to originate research ideas, generate experimental results, or assign scores. The evaluated agents are LLM systems, and their behavior is the subject of the experiments. Deterministic verifiers and evaluation code compute all benchmark scores. The description-fidelity judge (Appendix D) does not contribute to the progress score. The authors reviewed the AI-assisted work, checked every reported number against the released tables and data, checked mathematical statements against the cited sources, and ran generated code and inspected its outputs. We take responsibility for the final text, claims, and artifacts, including all AI-assisted content.

REFERENCES

- Mohammed Abouzaid, Andrew J. Blumberg, Martin Hairer, Joe Kileel, Tamara G. Kolda, Paul D. Nelson, Daniel Spielman, Nikhil Srivastava, Rachel Ward, Shmuel Weinberger, et al. First Proof. *arXiv preprint*, 2026. arXiv:2602.05192.
- Tudor Achim, Alex Best, Alberto Bietti, Kevin Der, Mathis Fédérico, Sergei Gukov, Daniel Halpern-Leistner, Kirsten Henningsgard, Yuri Kudryashov, Alexander Meiburg, et al. Aristotle: IMO-level automated theorem proving. *arXiv preprint*, 2025. arXiv:2510.01346.
- American Institute of Mathematics. AI and number theory. Workshop, 27–30 May 2026, <https://aimath.org/pastworkshops/aint26.html>, 2026. Accessed 2 October 2026.
- Chenyang An, Qihao Ye, Minghao Pan, and Jiayaun Zhang. QED: An open-source multi-agent system for generating mathematical proofs on open problems. *arXiv preprint*, 2026. arXiv:2604.24021.
- Benjamin Antieau. Jürgen Klüners and Gunter Malle on the inverse Galois problem and computation. Interview, <https://antieau.github.io/2026/08/27/gunter-malle-juergen-klueners-ai.html>, 2026. Published 27 August 2026; accessed 2 October 2026.
- Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Math-Arena: Evaluating LLMs on uncontaminated math competitions. *arXiv preprint*, 2025. arXiv:2505.23281.
- Gennadii V. Belyi. On Galois extensions of a maximal cyclotomic field. *Mathematics of the USSR-Izvestiya*, 14(2):247–256, 1980.
- Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Aleksander Madry, and Lilian Weng. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations*, 2025.

- François Charton, Jordan S. Ellenberg, Adam Zsolt Wagner, and Geordie Williamson. PatternBoost: Constructions in mathematics with a little help from AI. *arXiv preprint*, 2024. arXiv:2411.00566.
- Luoxin Chen, Jinming Gu, Liankai Huang, Wenhao Huang, Zhicheng Jiang, Allan Jie, Xiaoran Jin, Xing Jin, Chenggang Li, Kaijing Ma, et al. Seed-Prover: Deep and broad reasoning for automated theorem proving. *arXiv preprint*, 2025a. arXiv:2507.23726.
- Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, et al. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. *arXiv preprint*, 2023. arXiv:2308.10848.
- Xiaoyang Chen and Xiang Jiang. Moonshine: An autonomous mathematical research agent centered on conjecture generation. *arXiv preprint*, 2026. arXiv:2606.10806.
- Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do NOT think that much for $2+3=?$ on the overthinking of long reasoning models. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025b. arXiv:2412.21187.
- Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, et al. ScienceAgentBench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, 2025c.
- Lyndie Chiou. Mathematicians use AI to find mysterious symmetries, solving decades-old problem. *Scientific American*, September 2026. 22 September 2026, <https://www.scientificamerican.com/article/mathematicians-use-ai-to-find-mysterious-symmetries-solving-decades-old-problem/>.
- Kevin Coombes and David Harbater. Hurwitz families and arithmetic Galois groups. *Duke Mathematical Journal*, 52(4):821–839, 1985.
- Jean-Marc Couveignes. Tools for the computation of families of coverings. In *Aspects of Galois Theory*, volume 256 of *London Mathematical Society Lecture Note Series*, pp. 38–65. Cambridge University Press, 1999.
- Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, Nicholas Thumiger, Aditya Desai, Ion Stoica, Ana Klimovic, Graham Neubig, and Joseph E. Gonzalez. The danger of overthinking: Examining the reasoning-action dilemma in agentic tasks. In *Conference on Language Modeling*, 2025. arXiv:2502.08235.
- Alex Davies, Petar Veličković, Lars Buesing, Sam Blackwell, Daniel Zheng, Nenad Tomašev, Richard Tanburn, Peter Battaglia, Charles Blundell, András Juhász, Marc Lackenby, Geordie Williamson, Demis Hassabis, and Pushmeet Kohli. Advancing mathematics by guiding human intuition with AI. *Nature*, 600:70–74, 2021.
- Pierre Dèbes and Jean-Claude Douai. Algebraic covers: field of moduli versus field of definition. *Annales Scientifiques de l’École Normale Supérieure*, 30(3):303–338, 1997.
- Pierre Dèbes and Michael D. Fried. Nonrigid constructions in Galois theory. *Pacific Journal of Mathematics*, 163(1):81–122, 1994.
- Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint*, 2023. arXiv:2305.14325.
- Epoch AI. The inverse Galois problem for the Mathieu group M_{23} . FrontierMath: Open Problems, <https://epoch.ai/frontiermath/open-problems/inverse-galois>, 2026. Listed as unsolved by January 2026; marked “solved (human + AI)” as of 25 September 2026.
- Moran Feldman and Amin Karbasi. Gödel test: Can large language models solve easy conjectures? *arXiv preprint*, 2025. arXiv:2509.18383.

- Tony Feng, Trieu H. Trinh, Garrett Bingham, Dawsen Hwang, Yuri Chervonyi, Junehyuk Jung, Joonkyung Lee, Carlo Pagano, Sang-hyun Kim, Federico Pasqualotto, et al. Towards autonomous mathematics research. *arXiv preprint*, 2026. arXiv:2602.10177.
- Michael D. Fried. Fields of definition of function fields and Hurwitz families—groups as Galois groups. *Communications in Algebra*, 5(1):17–82, 1977.
- Michael D. Fried and Helmut Völklein. The inverse Galois problem and rational points on moduli spaces. *Mathematische Annalen*, 290:771–800, 1991.
- Suhaas Garre, Erik Knutsen, Sushant Mehta, and Edwin Chen. Riemann-Bench: A benchmark for moonshot mathematics. *arXiv preprint*, 2026. arXiv:2604.06802.
- Aryo Pradipta Gema, Alexander Hägele, Runjin Chen, Andy Ardit, Jacob Goldman-Wetzler, Kit Fraser-Taliente, Henry Sleight, Linda Petrini, Julian Michael, Beatrice Alex, Pasquale Minervini, Yanda Chen, Joe Benton, and Ethan Perez. Inverse scaling in test-time compute. *arXiv preprint*, 2025. arXiv:2507.14417.
- Bogdan Georgiev, Javier Gómez-Serrano, Terence Tao, and Adam Zsolt Wagner. Mathematical exploration and discovery at scale. *arXiv preprint*, 2025. arXiv:2511.02864.
- Elliot Glazer, Ege Erdil, Tamay Besiroglu, Diego Chicharro, Evan Chen, Alex Gunning, Caroline Falkman Olsson, Jean-Stanislas Denain, Anson Ho, Emily de Oliveira Santos, Olli Järvinen, Matthew Barnett, Robert Sandler, Matej Vrzala, Jaime Sevilla, Qiuyu Ren, Elizabeth Pratt, Lionel Levine, Grant Barkley, Natalie Stewart, Bogdan Grechuk, Tetiana Grechuk, Shreepranav Varma Enugandla, and Mark Wildon. FrontierMath: A benchmark for evaluating advanced mathematical reasoning in AI. *arXiv preprint*, 2024. arXiv:2411.04872.
- Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, Artiom Myaskovsky, Grzegorz Glowaty, Felix Weissenberger, Alessio Orlandi, Dan Popovici, et al. Accelerating scientific discovery with Co-Scientist. *arXiv preprint*, 2025. arXiv:2502.18864.
- Louis Granboulan. Construction d’une extension régulière de $\mathbb{Q}(T)$ de groupe de Galois M_{24} . *Experimental Mathematics*, 5(1):3–14, 1996.
- Ryan Greenblatt, Ajeya Cotra, and Hjalmar Wijk. Brief independent investigation of agents’ behavior, reasoning and collaboration in the OpenAI/Hugging Face hacking incident. METR, <https://metr.org/blog/2026-08-26-openai-hugging-face-incident-investigation/>, 2026. Published 26 August 2026.
- Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiwu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations*, 2024.
- Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MAgentBench: Evaluating language agents on machine learning experimentation. *arXiv preprint*, 2023. arXiv:2310.03302.
- Xiaoyu Huang, Blake Jackson, Kyu-Hwan Lee, Bjorn Poonen, Rachel Pries, and Shaowu Zhang. The Mathieu group M_{23} is a Galois group over $\mathbb{Q}$. *arXiv preprint*, 2026. arXiv:2608.08538.
- Thomas Hubert, Rishi Mehta, Laurent Sartran, et al. Olympiad-level formal mathematical reasoning with reinforcement learning. *Nature*, 2025. doi: 10.1038/s41586-025-09833-y.
- Hugging Face. Anatomy of a frontier lab agent intrusion: A technical timeline of the July 2026 incident. <https://huggingface.co/blog/agent-intrusion-technical-timeline>, 2026. Published 27 July 2026.
- Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024.

- Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott 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 Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know. *arXiv preprint*, 2022. arXiv:2207.05221.
- Sayash Kapoor, Benedikt Stroebl, Zachary S. Siegel, Nitya Nadgir, and Arvind Narayanan. AI agents that matter. *Transactions on Machine Learning Research*, 2025. arXiv:2407.01502.
- Michael Klug, Michael Musty, Sam Schiavone, and John Voight. Numerical calculation of three-point branched covers of the projective line. *LMS Journal of Computation and Mathematics*, 17(1):379–430, 2014.
- Patrick Tser Jern Kon, Qiuyi Ding, Jiachen Liu, Xinyi Zhu, Jingjia Peng, Jiarong Xing, Yibo Huang, Yiming Qiu, Jayanth Srinivasa, Myungjin Lee, Mosharaf Chowdhury, Matei Zaharia, and Ang Chen. EXP-Bench: Can AI conduct AI research experiments? In *International Conference on Learning Representations*, 2026.
- Thomas Kwa, Ben West, Joel Becker, Amy Deng, Katharyn Garcia, Max Hasin, Sami Jawhar, Megan Kinniment, Nate Rush, Sydney Von Arx, et al. Measuring AI ability to complete long software tasks. *arXiv preprint*, 2025. arXiv:2503.14499.
- Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for “mind” exploration of large language model society. In *Advances in Neural Information Processing Systems*, volume 36, 2023.
- Yong Lin, Shange Tang, Bohan Lyu, Ziran Yang, Jui-Hui Chung, Haoyu Zhao, Lai Jiang, Yihan Geng, Jiawei Ge, Jingruo Sun, et al. Goedel-Prover-V2: Scaling formal theorem proving with scaffolded data synthesis and self-correction. *arXiv preprint*, 2025. arXiv:2508.03613.
- Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024.
- Xun Liu, Yi Evie Zhang, Vira Kasprova, Parisa Rabbani, Pardis Sadat Zahraei, Tianyu Zhang, Ali Ebrahimpour-Borojeny, and Varun Chandrasekaran. AgentAbstain: Do LLM agents know when not to act? *arXiv preprint*, 2026. arXiv:2607.10059.
- 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*, 2024. arXiv:2408.06292.
- Han Luo, Bingbing Wen, and Lucy Lu Wang. Agentic abstention: Do agents know when to stop instead of act? *arXiv preprint*, 2026. arXiv:2606.28733.
- Gunter Malle. Polynomials with Galois groups $\text{Aut}(M_{22})$, M_{22} , and $\text{PSL}_3(\mathbb{F}_4) \cdot 2_2$ over $\mathbb{Q}$. *Mathematics of Computation*, 51(184):761–768, 1988.
- Gunter Malle and B. Heinrich Matzat. *Inverse Galois Theory*. Springer Monographs in Mathematics. Springer, 2nd edition, 2018.
- B. Heinrich Matzat. *Konstruktive Galoistheorie*, volume 1284 of *Lecture Notes in Mathematics*. Springer, 1987.
- B. Heinrich Matzat and Andreas Zeh-Marschke. Realisierung der Mathieugruppen M_{11} und M_{12} als Galoisgruppen über $\mathbb{Q}$. *Journal of Number Theory*, 23(2):195–202, 1986.
- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint*, 2025. arXiv:2501.19393.

- Fan Nie, Ken Ziyu Liu, Zihao Wang, Rui Sun, Wei Liu, Weijia Shi, Huaxiu Yao, Linjun Zhang, Andrew Y. Ng, James Zou, et al. UQ: Assessing language models on unsolved questions. *arXiv preprint*, 2025. arXiv:2508.17580.
- Alexander Novikov, Ngán Vū, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. AlphaEvolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint*, 2025. arXiv:2506.13131.
- OpenAI. The Hugging Face incident and the road ahead. <https://openai.com/index/hugging-face-incident-and-the-road-ahead/>, 2026a. Published 26 August 2026.
- OpenAI. Safety and alignment in an era of long-horizon models. <https://openai.com/index/safety-alignment-long-horizon-models/>, 2026b. Published 20 July 2026.
- QuocViet Pham, Elvir Karimov, Andrey Galichin, and Ivan Oseledets. TheoremBench: Evaluating LLMs on theorem proving in formal mathematics. *arXiv preprint*, 2026. arXiv:2606.09450.
- Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity’s Last Exam. *arXiv preprint*, 2025. arXiv:2501.14249.
- Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. ChatDev: Communicative agents for software development. *arXiv preprint*, 2023. arXiv:2307.07924.
- Z. Z. Ren, Zhihong Shao, Junxiao Song, Huajian Xin, Haocheng Wang, Wanjia Zhao, Liyue Zhang, Zhe Fu, Qihao Zhu, Dejian Yang, et al. DeepSeek-Prover-V2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. *arXiv preprint*, 2025. arXiv:2504.21801.
- Matthieu Romagny and Stefan Wewers. Hurwitz spaces. In *Groupes de Galois arithmétiques et différentiels*, volume 13 of *Séminaires et Congrès*, pp. 313–341. Société Mathématique de France, 2006.
- Bernardino Romera-Paredes, Mohammadamin Barekatin, 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*, 625:468–475, 2024.
- SAIR Foundation. Inverse Galois problem (IGP24). <https://competition.sair.foundation/competitions/igp24/overview>, 2026. Accessed 2 October 2026.
- Johannes Schmitt, Gergely Bérczi, Jasper Dekoninck, Jeremy Feusi, Tim Gehringer, Raphael Appenzeller, Jim Bryan, Niklas Canova, Timo de Wolff, Filippo Gaia, Michel van Garrel, Baran Hashemi, David Holmes, Aitor Iribar Lopez, Victor Jaeck, Martina Jørgensen, Steven Kelk, Stefan Kuhlmann, Adam Kurpisz, Chiara Meroni, Ingmar Metzler, Martin Möller, Samuel Muñoz-Echániz, Robert Nowak, Georg Oberdieck, Daniel Platt, Dylan Possamai, Gabriel Ribeiro, Raúl Sánchez Galán, Zheming Sun, Josef Teichmann, Richard P. Thomas, and Charles Vial. IMProofBench: Benchmarking AI on research-level mathematical proof generation. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2025. arXiv:2509.26076.
- Jean-Pierre Serre. *Topics in Galois Theory*, volume 1 of *Research Notes in Mathematics*. Jones and Bartlett, 1992.
- Zhihong Shao, Yuxiang Luo, Chengda Lu, Z. Z. Ren, Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, and Xiaokang Zhang. DeepSeekMath-V2: Towards self-verifiable mathematical reasoning. *arXiv preprint*, 2025. arXiv:2511.22570.

- Zachary S. Siegel, Sayash Kapoor, Nitya Nadgir, Benedikt Stroebl, and Arvind Narayanan. CORE-Bench: Fostering the credibility of published research through a computational reproducibility agent benchmark. *arXiv preprint*, 2024. arXiv:2409.11363.
- Jeroen Sijsling and John Voight. On computing Belyi maps. *Publications Mathématiques de Besançon. Algèbre et Théorie des Nombres*, pp. 73–131, 2014.
- Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint*, 2024. arXiv:2408.03314.
- Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. PaperBench: Evaluating AI’s ability to replicate AI research. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, 2025.
- Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. Stop overthinking: A survey on efficient reasoning for large language models. *Transactions on Machine Learning Research*, 2025. arXiv:2503.16419.
- John G. Thompson. Some finite groups which appear as $\text{Gal } L/K$, where $K \subseteq \mathbb{Q}(\mu_n)$. *Journal of Algebra*, 89(2):437–499, 1984.
- Trieu H. Trinh, Yuhuai Wu, Quoc V. Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625:476–482, 2024.
- George Tsoukalas, Jasper Lee, John Jennings, Jimmy Xin, Michelle Ding, Michael Jennings, Amityay Thakur, and Swarat Chaudhuri. PutnamBench: Evaluating neural theorem-provers on the Putnam mathematical competition. *arXiv preprint*, 2024. arXiv:2407.11214.
- Helmut Völklein. *Groups as Galois Groups: An Introduction*, volume 53 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 1996.
- Adam Zsolt Wagner. Constructions in combinatorics via neural networks. *arXiv preprint*, 2021. arXiv:2104.14516.
- Erik Y. Wang, Sumeet Motwani, James V. Roggeveen, Eliot Hodges, Dulhan Jayalath, Charles London, Kalyan Ramakrishnan, Flaviu Cipcigan, Philip Torr, and Alessandro Abate. Measuring progress in reasoning toward mathematical discovery with automatic verification. *arXiv preprint*, 2026. arXiv:2603.15617.
- Haiming Wang, Mert Unsal, Xiaohan Lin, Mantas Baksys, Junqi Liu, Marco Dos Santos, Flood Sung, Marina Vinyes, Zhenzhe Ying, Zekai Zhu, et al. Kimina-Prover preview: Towards large formal reasoning models with reinforcement learning. *arXiv preprint*, 2025a. arXiv:2504.11354.
- Yue Wang, Qiuzhi Liu, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Linfeng Song, Dian Yu, Juntao Li, Zhuosheng Zhang, et al. Thoughts are all over the place: On the underthinking of o1-like LLMs. *arXiv preprint*, 2025b. arXiv:2501.18585.
- Bingbing Wen, Jihan Yao, Shangbin Feng, Chenjun Xu, Yulia Tsvetkov, Bill Howe, and Lucy Lu Wang. Know your limits: A survey of abstention in large language models. *arXiv preprint*, 2024. arXiv:2407.18418.
- Hjalmar Wijk, Tao Lin, Joel Becker, Sami Jawhar, Neev Parikh, Thomas Broadley, Lawrence Chan, Michael Chen, Josh Clymer, Jai Dhyani, et al. RE-Bench: Evaluating frontier AI R&D capabilities of language model agents against human experts. *arXiv preprint*, 2024. arXiv:2411.15114.
- Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint*, 2023. arXiv:2308.08155.

- Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist-v2: Workshop-level automated scientific discovery via agentic tree search. *arXiv preprint*, 2025. arXiv:2504.08066.
- Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic early exit in reasoning models. *arXiv preprint*, 2025. arXiv:2504.15895.
- Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ -bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint*, 2024. arXiv:2406.12045.
- Jie Zhang, Cezara Petrucci, Kristina Nikolić, and Florian Tramèr. RealMath: A continuous benchmark for evaluating language models on research-level mathematics. *arXiv preprint*, 2025. arXiv:2505.12575.
- Shaowu Zhang. Where to point the telescope: AI, mathematical judgment, and M_{23} . Proofs and Prompts, <https://proofsandprompts.com/2026/08/21/where-to-point-the-telescope/>, 2026. Published 21 August 2026.
- Daniel Zheng, Ingrid von Glehn, Yori Zwols, Iuliya Beloshapka, Lars Buesing, Daniel M. Roy, Martin Wattenberg, Bogdan Georgiev, Tatiana Schmidt, Andrew Cowie, et al. AI co-mathematician: Accelerating mathematicians with agentic AI. *arXiv preprint*, 2026. arXiv:2605.06651.
- Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. MiniF2F: a cross-system benchmark for formal olympiad-level mathematics. In *International Conference on Learning Representations*, 2022.
- David Zywina. The inverse Galois problem for $\mathrm{PSL}_2(\mathbb{F}_p)$. *Duke Mathematical Journal*, 164(12): 2253–2292, 2015.

A ADDITIONAL RELATED WORK

Mathematics benchmarks. Mathematics benchmarks range from competition and formal problems, such as miniF2F, PutnamBench, and MathArena (Zheng et al., 2022; Tsoukalas et al., 2024; Balunović et al., 2025), to expert- and research-level questions, such as Humanity’s Last Exam, RealMath, First Proof, UQ, and the Gödel Test (Phan et al., 2025; Zhang et al., 2025; Abouzaid et al., 2026; Nie et al., 2025; Feldman & Karbasi, 2025).

AI systems for mathematics. Machine learning has guided mathematical intuition (Davies et al., 2021) and found new constructions through search, as in FunSearch, AlphaEvolve, and PatternBoost (Wagner, 2021; Romera-Paredes et al., 2024; Novikov et al., 2025; Georgiev et al., 2025; Charton et al., 2024). AlphaGeometry, AlphaProof, Aristotle, Seed-Prover, DeepSeek-Prover-V2, Kimina-Prover, Goedel-Prover-V2, and DeepSeekMath-V2 target competition-level and formal theorem proving (Trinh et al., 2024; Hubert et al., 2025; Achim et al., 2025; Chen et al., 2025a; Ren et al., 2025; Wang et al., 2025a; Lin et al., 2025; Shao et al., 2025). Research agents such as Aletheia, the AI co-mathematician, QED, and Moonshine work on research-level mathematics (Feng et al., 2026; Zheng et al., 2026; An et al., 2026; Chen & Jiang, 2026).

Research agents and multi-agent systems. MLE-bench, RE-Bench, MAgentBench, ScienceAgentBench, CORE-Bench, and SWE-bench evaluate agents on engineering and scientific tasks (Chan et al., 2025; Wijk et al., 2024; Huang et al., 2023; Chen et al., 2025c; Siegel et al., 2024; Jimenez et al., 2024). AgentBench and τ -bench on general tool use (Liu et al., 2024; Yao et al., 2024), and Kwa et al. (2025) measure the length of tasks agents can complete. The AI Scientist, the AI Scientist-v2, and the AI co-scientist automate parts of the research process (Lu et al., 2024; Yamada et al., 2025; Gottweis et al., 2025). ChatDev and AgentVerse assign agents different roles, and multi-agent debate lets agents critique one another (Qian et al., 2023; Chen et al., 2023; Du et al., 2023).

Reasoning length and abstention. Reasoning models can also underthink (Wang et al., 2025b), and budget forcing and early exit control how long a model reasons (Muennighoff et al., 2025; Yang et al., 2025). Language models partly know what they know (Kadavath et al., 2022), and abstention has been studied for language models and agents (Wen et al., 2024; Luo et al., 2026).

B MATHEMATICAL BACKGROUND AND THE CONSTRUCTION PIPELINE

B.1 MATHEMATICAL BACKGROUND

Regular realizations. Fix a finite group G and write $G_{\mathbb{Q}} = \text{Gal}(\overline{\mathbb{Q}}/\mathbb{Q})$. A *regular realization* of G over $\mathbb{Q}(T)$ is a Galois extension $N/\mathbb{Q}(T)$ with group G in which $\mathbb{Q}$ is algebraically closed. It is given by a polynomial $F(T, V) \in \mathbb{Q}[T, V]$ with splitting field N . By Hilbert’s irreducibility theorem, $F(t_0, V)$ has Galois group G over $\mathbb{Q}$ for infinitely many $t_0 \in \mathbb{Q}$ (Serre, 1992; Völklein, 1996; Malle & Matzat, 2018). Specializing a regular realization is the classical way to realize a simple group over $\mathbb{Q}$ and the one used for all 26 sporadic groups (Malle & Matzat, 2018; Huang et al., 2026); other methods exist, for example Galois representations for $\text{PSL}_2(\mathbb{F}_p)$ (Zywina, 2015).

Covers and Nielsen classes. A regular realization is the function field of a G -cover of $\mathbb{P}^1$ defined over $\mathbb{Q}$, that is, of a Galois cover with group G that is defined over $\mathbb{Q}$ together with its automorphisms. Over $\mathbb{C}$, Riemann’s existence theorem classifies the G -covers with given branch points $t_1, \dots, t_r$: after a choice of standard generators of $\pi_1(\mathbb{P}^1 \setminus \{t_1, \dots, t_r\})$, they correspond to the tuples $(g_1, \dots, g_r)$ of non-identity elements of G with $g_1 \cdots g_r = 1$ and $\langle g_1, \dots, g_r \rangle = G$, up to simultaneous conjugation. Such a tuple is a *generating tuple*. For a *class vector* $C = (C_1, \dots, C_r)$ of conjugacy classes, the *Nielsen class* $\text{Ni}(G; C)$ is the set of generating tuples with $g_i \in C_i$, up to simultaneous conjugation, and $|\text{Ni}(G; C)|$ is the *Nielsen count*. The braid group on r strands acts on the larger set in which the classes $C_1, \dots, C_r$ occur in any order. Its orbits, the *braid orbits*, correspond to the connected components of the Hurwitz space $\mathcal{H}(G; C)$ of G -covers with r unordered branch points and these classes (Fried, 1977; Fried & Völklein, 1991; Romagny & Wewers, 2006). The pure braid group preserves $\text{Ni}(G; C)$, and its orbits there correspond to the components for or-

dered branch points. For $r = 3$ the pure braid group acts trivially: each element of $\text{Ni}(G; C)$ is one cover, and a braid orbit is one cover with its branch points in every order.

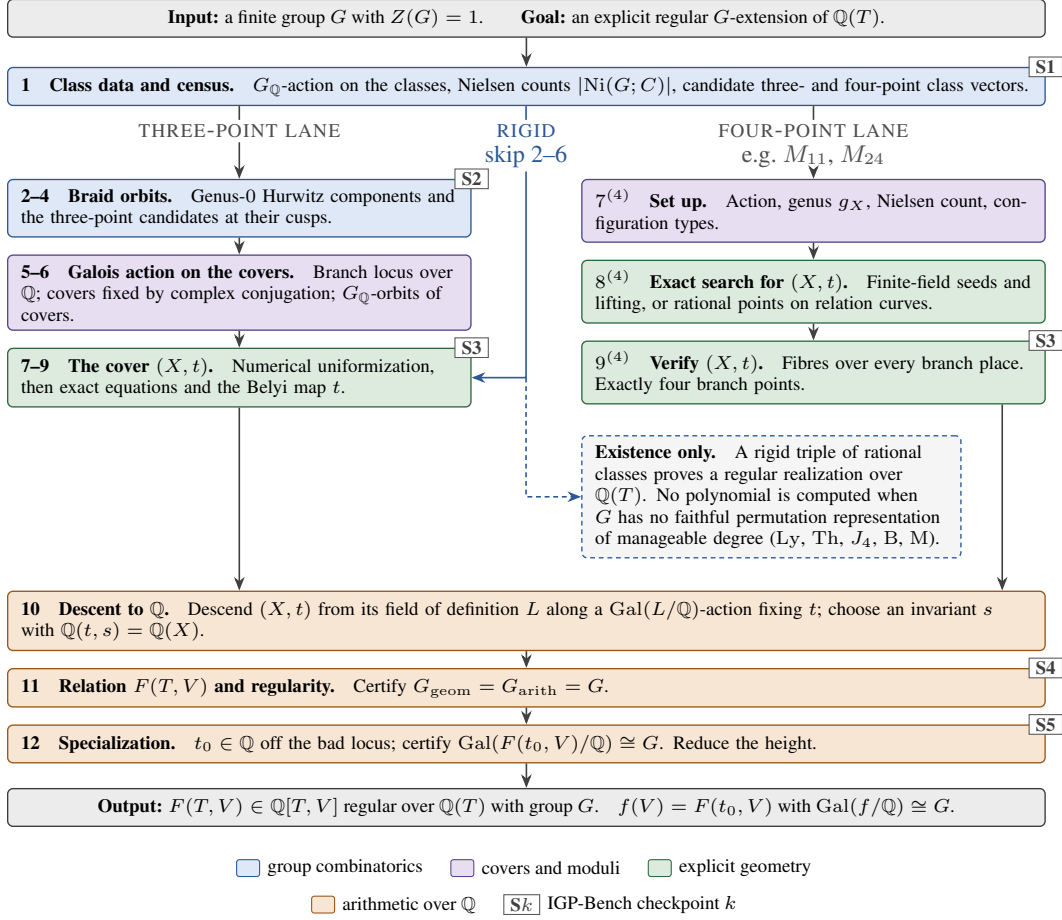


Figure 5: **The construction pipeline** followed by GALOIS. Stage 1 is common to all groups. The pipeline then splits by the number of branch points and rejoins at Stage 10; the descent to $\mathbb{Q}$ and the certification are the same in both lanes. Only Stage 7 of the three-point lane is numerical. G_{arith} and G_{geom} are the Galois groups of F over $\mathbb{Q}(T)$ and over $\overline{\mathbb{Q}}(T)$. A rigid triple of rational classes ($|\text{Ni}(G; C)| = 1$) skips Stages 2–6; it proves existence, and it yields a polynomial only if the group has a faithful permutation representation of manageable degree. White tags mark the five generic IGP-Bench checkpoints: S1 generating tuple, S2 braid orbit, S3 geometric cover, S4 regular family over $\mathbb{Q}(T)$, and S5 final polynomial over $\mathbb{Q}$.

Necessary conditions over $\mathbb{Q}$. Let $\chi: G_{\mathbb{Q}} \rightarrow (\mathbb{Z}/|G|\mathbb{Z})^{\times}$ be the cyclotomic character, so that $\sigma(\zeta) = \zeta^{\chi(\sigma)}$ for every $|G|$ -th root of unity ζ , and write C^m for the class of the m -th powers of the elements of a class C . A class vector is *Galois-stable* if, as a multiset, it is stable under $C_i \mapsto C_i^m$ for every m prime to $|G|$. Suppose that a G -cover with class C_i at t_i is defined over $\mathbb{Q}$. Then (i) $G_{\mathbb{Q}}$ permutes the branch points, and the class at $\sigma(t_i)$ is $C_i^{\chi(\sigma)^{-1}}$ (the branch cycle lemma; Fried, 1977), so that C is Galois-stable; (ii) the cover is a $\mathbb{Q}$ -rational point of $\mathcal{H}(G; C)$, which is defined over $\mathbb{Q}$ because C is Galois-stable, so the component that contains it, that is, its braid orbit, is defined over $\mathbb{Q}$; (iii) its element of $\text{Ni}(G; C)$ is fixed by complex conjugation, which acts on the Nielsen class by an explicit involution determined by which branch points are real (Dèbes & Fried, 1994). Conditions (i) and (iii) are finite computations in G ; condition (ii) can be read off the group only in special cases. The simplest case of (i) is that of rational classes at rational branch points. For $(2A, 23A, 23B)$ in M_{23} , the classes $23A$ and $23B$ are interchanged by $\text{Gal}(\mathbb{Q}(\sqrt{-23})/\mathbb{Q})$, so the two branch points that carry them lie in $\mathbb{Q}(\sqrt{-23})$ and are conjugate. Conditions (i) and (iii)

are necessary, not sufficient: exactly one of the seven covers of this class vector is defined over $\mathbb{Q}$ (Huang et al., 2026), while complex conjugation fixes three of them.

Rigidity and explicit construction. Suppose that $Z(G) = 1$. A class vector C is *rigid* if $|\text{Ni}(G; C)| = 1$. If C is rigid and its classes are rational, then G has a regular realization over $\mathbb{Q}(T)$ with rational branch points, and no equation needs to be computed (Belyi, 1980; Thompson, 1984; Matzat, 1987). This theorem and its generalizations give regular realizations of every sporadic group except M_{23} (Malle & Matzat, 2018). An explicit polynomial requires equations for the cover; among the Mathieu groups they were published for M_{11} , M_{12} , M_{22} , and M_{24} (Matzat & Zeh-Marschke, 1986; Malle, 1988; Granboulan, 1996) and, in 2026, for M_{23} (Huang et al., 2026). In the pipeline of Figure 5, the equations of a three-point cover are found by computing a numerical uniformization, recognizing exact equations, and descending them to $\mathbb{Q}$ (Couveignes, 1999; Klug et al., 2014; Sijsling & Voight, 2014); four-point covers are found by an exact search.

B.2 PIPELINE STAGES

Figure 5 divides the construction into twelve stages, which GALOIS follows, and marks the five generic IGP-Bench checkpoints. Appendix C goes through the stages for M_{23} .

C THE M_{23} TASK

The worked route. The twelve steps below follow the stages of Figure 5 for the realization of M_{23} (Huang et al., 2026; Zhang, 2026). The correct route is the three-point class vector $(2A, 23A, 23B)$. Each of its seven covers also arises at the cusps of one of seven pure braid orbits of length two for the four-point class vector $(23A, 23A, 23B, 23B)$, and steps 2–4 reach the route in this way.

1. **Candidates.** M_{23} has 16 nontrivial conjugacy classes, hence $\binom{18}{3} = 816$ unordered class triples, of which 86 are Galois-stable. No Galois-stable triple has Nielsen count one, so rigidity does not apply. Among the Galois-stable triples, the smallest positive Nielsen count is 7, attained by $(2A, 23A, 23B)$ (Huang et al., 2026). The benchmark also records routes that were tried in the literature and failed, such as $(2A, 8A, 8A)$.
2. **Braid orbits.** Each pair $a \in 23A$, $b \in 23B$ with $ab \in 2A$ and $\langle a, b \rangle = M_{23}$ gives a seed (a, a, b, aba^{-1}) in $\text{Ni}(M_{23}; 23A, 23A, 23B, 23B)$. There are seven seeds. Closing each of them under the pure braid generators $\beta_{12}, \beta_{13}, \beta_{14}$ in GAP gives seven distinct orbits $Z_1, \dots, Z_7$, each of length two.
3. **Genus and cusps.** Let λ be the cross-ratio of the four branch points. Each Z_i is the fibre of a cover of the λ -line of degree $|Z_i| = 2$, unramified outside $0, 1, \infty$. The three braid generators act on Z_i with $(c_1, c_2, c_3) = (2, 1, 1)$ cycles, so Riemann–Hurwitz gives $g = 1 + \frac{1}{2}(|Z_i| - c_1 - c_2 - c_3) = 1 + \frac{1}{2}(2 - 4) = 0$. The two cusps over one of the three points carry triples in $(23A, 23B, 23B)$ and in $(23A, 23A, 23B)$; the cusp over each of the other two points carries the triple (ab, a, b) of the seed, in $(2A, 23A, 23B)$, so each of the seven covers of the route arises from exactly one Z_i .
4. **Route identification.** Check that each orbit contains a seed (a, a, b, aba^{-1}) with $\langle a, b \rangle = M_{23}$, that $(2A, 23A, 23B)$ is Galois-stable, and that the number of orbits equals $|\text{Ni}(M_{23}; 2A, 23A, 23B)| = 7$.
5. **Branch points.** The classes $23A$ and $23B$ are interchanged by $\text{Gal}(\mathbb{Q}(\sqrt{-23})/\mathbb{Q})$. Take the branch locus $T(T^2 + 23) = 0$, with $2A$ at 0 and $23A, 23B$ at $\pm\sqrt{-23}$.
6. **Galois action on the covers.** $G_{\mathbb{Q}}$ permutes the seven covers, with orbits of lengths 1 and 6 (Huang et al., 2026); complex conjugation acts by $[(g_1, g_2, g_3)] \mapsto [(g_1^{-1}, g_3^{-1}, g_2^{-1})]$ and fixes three of them. The cover fixed by $G_{\mathbb{Q}}$ has field of moduli $\mathbb{Q}$. Since M_{23} has trivial center and is its own normalizer in S_{23} , the field of moduli is a field of definition (Coombes & Harbater, 1985; Dèbes & Douai, 1997), so the cover descends to $\mathbb{Q}$.
7. **Analytic model.** The curve X is non-hyperelliptic of genus 4. A uniformization $X(\mathbb{C}) \cong \Gamma \backslash \mathbb{D}$ by a Fuchsian group Γ acting on the unit disc $\mathbb{D}$ gives four holomorphic differentials as power series at a suitable point.

8. **Exact equations.** The four series define the canonical map to $\mathbb{P}^3$, whose image is cut out by a quadric Φ_2 and a cubic Φ_3 with coefficients in $\mathbb{Q}(\sqrt{-23})$. Both must be recognized from numerical data and then checked exactly. In triangle-group coordinates, the recognition failed at 40, 60, and 90 digits. In Galois-equivariant coordinates defined through the Belyi map, 40 digits sufficed for Φ_2 , and a separate reduction algorithm recovered Φ_3 from the same data (Zhang, 2026).
9. **The map.** Recover the degree-23 map $t: X \rightarrow \mathbb{P}^1$ from the points above the branch locus.
10. **Descent.** Descend (X, t) to $\mathbb{Q}$ and choose $s \in \mathbb{Q}(X)$ with $\mathbb{Q}(X) = \mathbb{Q}(t, s)$.
11. **Relation and regularity.** Compute $F(T, V) \in \mathbb{Z}[T, V]$ of degree 8 in T and 23 in V with $F(t, s) = 0$. Certify that F is irreducible with the expected branch behavior and that $G_{\text{geom}} = G_{\text{arith}} = M_{23}$; the identification of the group uses a reduction modulo 31 checked in Magma.
12. **Specialization.** Choose a rational t_0 off the bad locus, set $f(V) = F(t_0, V)$, reduce f with `polredabs`, and certify $\text{Gal}(f/\mathbb{Q}) \cong M_{23}$ in Magma. The number field of the published polynomial is unramified outside $\{2, 3, 23\}$.

Checkpoint ladder. The M_{23} task has seven checkpoints instead of the generic five. The construction score C is the value attached to the deepest verified checkpoint; the generic checkpoint to which each one corresponds is given in parentheses.

1. A generating tuple of M_{23} , given by permutations of degree 23 (S1; $C = 0$).
2. A complete list of representatives of the elements of the Nielsen class of one class vector, enumerated in GAP under a cap of 2×10^6 candidate tuples (S2; $C = 5$).
3. A geometric cover: an equation over $\mathbb{Q}$ with its branch cycles, or an equation over a number field, optionally with data that identify its branch cycles (S3; $C = 15$).
4. A certificate that the field of moduli is $\mathbb{Q}$ (S4; $C = 20$).
5. A descended cover over $\mathbb{Q}$ with its descent isomorphisms (S4; $C = 25$).
6. A regular family over $\mathbb{Q}(T)$, verified from exact branch data and from the Galois group, M_{23} , of its reduction modulo a prime of good reduction (S4; $C = 30$). Magma’s generic computation of the geometric Galois group does not finish in practical time in degree 23.
7. A degree-23 polynomial that is irreducible, has square discriminant, and has Galois group M_{23} by Magma’s `GaloisProof`, or a specialization certificate for a verified family (S5; $C = 40$).

The route-identification score has a task-specific rule: $R = 30$ for a verified braid-orbit size, a verified Nielsen count, or a complete list of representatives of the elements of $\text{Ni}(M_{23}; 2A, 23A, 23B)$, attached to an explicit generating triple. Checkpoint 2, by contrast, accepts a complete list of representatives for any class vector, which is why 16 of the 18 runs on M_{23} obtain the 5 points of checkpoint 2 ($C = 12.5$ in Table 9) although only four of them earn R . The screening score also has a task-specific schedule. Verified facts have weights from 1, for a product-one check, to 6, for a Hurwitz model or an existence theorem. Within each of six categories of facts, a class vector counts once, with its heaviest verified result, so results on different class vectors add up, unlike on the generic ladder.

D METRIC DEFINITIONS

D.1 PROGRESS SCORE

Checkpoint ladders. The generic ladder has five checkpoints: a generating tuple (S1), a braid orbit (S2), a geometric cover (S3), a regular family over $\mathbb{Q}(T)$ (S4), and a final polynomial (S5). Six tasks, those for M_{11} , M_{23} , $\text{SL}_2(13)$, $\text{SL}_2(19)$, and two Rubik’s cube groups, have ladders adapted to their known constructions. Appendix C gives the M_{23} ladder; the M_{11} runs of this paper were scored on the generic ladder.

Screening score. S is the sum of the weights of the verified facts that the run establishes about class vectors, tuples, orbits, families, and polynomials, capped at 30. On the generic ladder, each kind of fact has a weight from 1 (a necessary condition) to 3 (rigidity) and counts once per run, on

however many class vectors it is verified; statistical evidence earns nothing. Task-specific schedules may use other weights and another counting rule (Appendix C). A run that earns the route-identification score receives $S = 30$.

Route-identification score. $R = 30$ if a single artifact contains a generating tuple on a correct route together with a verified braid-orbit computation on that route, and $R = 0$ otherwise. A task with its own ladder may have its own rule; Appendix C gives the rule for M_{23} .

Construction score. C is determined by the deepest construction checkpoint verified on a single route. On the generic ladder, a generating tuple and a braid orbit earn no construction points, a cover earns 15, a regular family 30, and a final polynomial 40. Construction credit does not require prior literature support for the route: a verified polynomial earns $C = 40$ whatever route produced it.

Route ledger. The ledger lists 141 routes from the literature, 129 of them for 64 of the benchmark’s groups. Section 3.2 defines which routes are correct for a task. For R , the evaluator identifies the conjugacy classes of a submitted tuple in GAP and compares them with the ledger; the class names that the agent declares play no role.

D.2 RESOURCE METRICS

Cost attribution. An agent that opens an artifact declares it either as *screening*, that is, work not dedicated to one class vector, or as *research* on one or more named class vectors; an artifact shared by several class vectors carries weights that sum to one. Work on a strategy that is not a class vector is declared as such and enters neither the screening cost nor a route cost. The evaluator attributes each model call and each tool call to the artifact that is open when the call is made. The agent finalizes an artifact as a result, a negative result, or abandoned, and an artifact that is open when the run ends is finalized as abandoned. We use three resources r : output tokens (the default), tool CPU time, and worker time. Output tokens are the provider’s output count; where a provider also reports reasoning tokens, they are part of that count. Tool CPU time excludes model inference. Worker time is the time during which the artifact is open, summed over parallel workers, so it can exceed the elapsed time. A resource that was not measured for a run is undefined for that run. Cost incurred while no artifact is open is reported separately as unattributed and enters neither U_r nor K_r , so a run that records only part of its work in artifacts is scored on that part.

Resource efficiency. Let A be the set of finalized artifacts and $r(v)$ the cost of $v \in A$. An artifact is *credited* if a checkpoint verified in it is deeper than every checkpoint verified earlier in the run, if the verified final submission is made from it, if it is the first artifact in which a fact counted in S is verified, or if it earns R . Let $U \subseteq A$ consist of the credited artifacts and of every artifact on which they depend, directly or indirectly. Then

$$U_r = 100 \frac{\sum_{v \in U} r(v)}{\sum_{v \in A} r(v)}.$$

Knowing when to stop. Let B_r be the total cost of the screening artifacts. A route is *researched* if some research artifact names it; here a route is a group together with class names as the agent declares them. Its cost is the sum of the costs of the research artifacts that name it, each multiplied by its weight for the route. Let n_+ and n_- be the numbers of researched correct and wrong routes and C_r^+ , C_r^- their total costs. Put $\bar{C}_r^\pm = C_r^\pm / n_\pm$ if $n_\pm > 0$ and $\bar{C}_r^\pm = 0$ otherwise. Equation (2) then defines K_r . In particular, $K_r = 100$ if only correct routes are researched, $K_r = 100 B_r / (B_r + \bar{C}_r^-)$ if only wrong routes are, and $K_r = 0$ if in addition the run records no screening cost. K_r is undefined if no route is researched or if the denominator is zero. A class vector that occurs only in screening artifacts contributes to B_r but to neither route count. Figures 6 and 7 show an example graph, on which both metrics are computed, and the artifact graph of a run.

Route labels. K_r requires a label, correct or wrong, for every researched route. The evaluator assigns these labels during scoring. We also label the researched routes afterward against the literature and apply the same formula; we call these *retrospective* labels. K_r uses the evaluator’s labels where they are recorded and retrospective labels otherwise (Table 8).

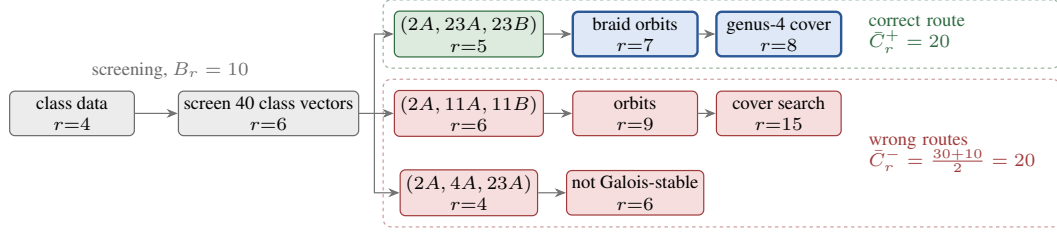


Figure 6: U_r and K_r on an example graph. Each artifact v is marked with its cost $r(v)$. Bold blue boxes mark the two credited artifacts, each with a verified checkpoint; no other artifact earns credit in this example. Grey: screening, $B_r = 10$. Green: the correct route, with research cost $\bar{C}_r^+ = 20$. Red: two wrong routes, with research costs 30 and 10, so $\bar{C}_r^- = 20$. The credited artifacts and the artifacts they depend on cost 30 out of 70, so $U_r = 100 \cdot 30/70 \approx 43$, and $K_r = 100(10 + 20)/(10 + 20 + 20) = 60$. A value $K_r = 100$ means that no cost is spent on wrong routes after screening.

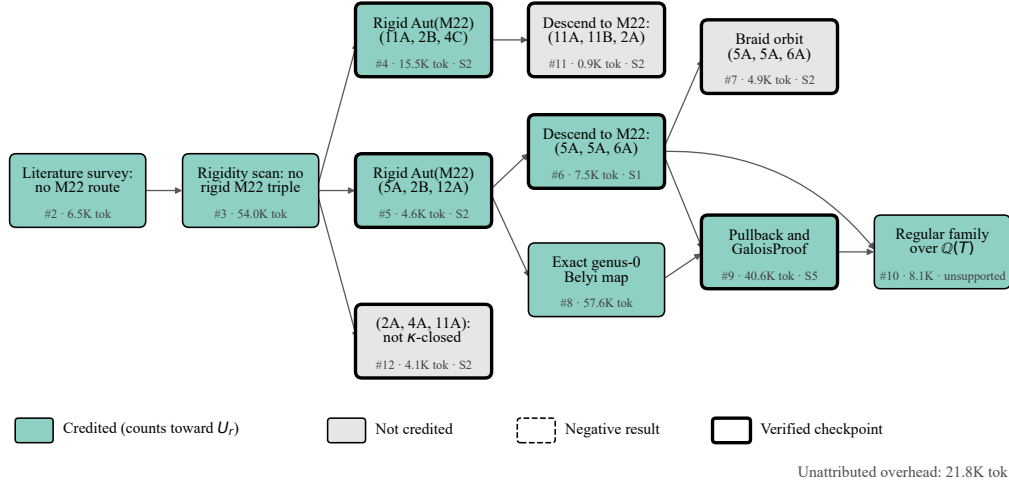


Figure 7: Artifact graph for a Claude Opus 5 run that realizes M_{22} ($P = 100$). Boxes are arranged by dependency depth and show output tokens and the strength of the certified checkpoint. Filled boxes count toward U_r ; bold borders mark verified checkpoint claims. The run finds no rigid triple of rational classes in M_{22} , certifies two rigid triples in $\text{Aut}(M_{22})$, descends one to M_{22} , solves its Belyi map exactly, and certifies the pulled-back polynomial (S5).

D.3 PROTOCOL AND TOOL-USE METRICS

Format compliance. A deterministic formatter checks each artifact record and each final submission against the published format, and issues at most two repair reminders. Let n_i be the number of records and submissions accepted after exactly i reminders and n_{fail} the number never accepted; a resubmission after a reminder is not counted again. Then

$$I_f = \frac{100 n_0 + 50 n_1 + \frac{100}{3} n_2}{n_0 + n_1 + n_2 + n_{\text{fail}}},$$

which is undefined for a run with no record and no submission.

Description fidelity. An LLM judge (Claude Sonnet 5, called through the Claude Code CLI with tools disabled) compares each finalized record of a result or of a negative result with the execution log and labels it correct, incorrect, or undetermined. If N_c records are labeled correct and N_i incorrect, then $\eta = 100 N_c / (N_c + N_i)$, which is undefined if $N_c + N_i = 0$. Description fidelity does not enter the progress score.

Tool use. Let T be the set of tool calls and $T_v \subseteq T$ the set of calls with a parseable action and valid arguments. *Validity* is $|T_v|/|T|$. *Termination* is the fraction of the calls in T_v that run to completion; a call that exits with a nonzero status has completed.

D.4 TASKS WITHOUT A KNOWN CORRECT ROUTE

Verification requires no reference solution, so screening and construction are also scored on tasks for which the ledger contains no correct route, including tasks with known realizations.

The *open-case profile* scores a task for which the ledger has no correct route (Table 11). It keeps the 30/30/40 split and the generic construction ladder, and it replaces route identification by the resolution of a candidate passport. Let n be the degree of the permutation representation of G fixed by the task. A *candidate passport* is a class vector $\{C_1, C_2, C_3\}$ of non-identity classes, taken up to order, such that (i) it is Galois-stable; (ii) at least $|G|/|Z(G)|$ triples $(g_1, g_2, g_3) \in C_1 \times C_2 \times C_3$ have product one, the number that one generating triple contributes; (iii) its Riemann–Hurwitz genus g in degree n satisfies $0 \leq g \leq g_{\max}$. We take $g_{\max} = 2$, except for $\text{Sz}(8)$ and $\text{PSL}_2(27)$, where $g_{\max}$ is the smallest genus $g \geq 0$ of a class vector that satisfies (i) and (ii), 4 and 9, respectively; for $\text{Sz}(32)$ no class vector with $0 \leq g \leq 12$ satisfies (i) and (ii), and $g_{\max} = 2$ is kept. Every regular realization of G over $\mathbb{Q}(T)$ with three branch points whose cover of degree n has genus at most $g_{\max}$ has its class vector among the candidates.

A run *resolves* a candidate by one of three results, the first two recomputed by the evaluator in GAP: no triple in (ii) generates G (N4); no element of $\text{Ni}(G; C)$ is fixed by complex conjugation, whether the three branch points are real or one is real and two are complex conjugate, so that no G -cover with this passport is defined over $\mathbb{R}$ (N7; Dèbes & Fried, 1994); or a verified cover over $\mathbb{Q}$ with this passport. The first artifact that resolves a candidate earns $R = 30$, once per run, and brings S to 30. In addition to the facts of the generic ladder, S credits a correct list of the candidates, correct counts of the elements of $\text{Ni}(G; C)$ and of those fixed by complex conjugation, and an obstruction N7. Numerical results earn no credit.

E EXPERIMENTAL SETUP DETAILS

Table 4 lists the nine systems.

Table 4: System configurations. Every run has the same budget: 10 hours, a cap of 500,000 output tokens, eight CPUs, and 64–192 GB of memory depending on the task. Each system has two runs per task; the two Fable runs per task come from separate batches, on 23 and 29 September 2026. Other runs, among them earlier 6-hour runs of GPT-5.6 Sol and Terra and the Claude Haiku runs, are not included.

System	Model id	Harness	Effort	Runs per task	n
Claude Opus 5	claude-opus-5	Claude native (Claude Code)	xhigh	2	10
GPT-6 Sol	gpt-6-sol	Codex native	xhigh	2	10
GPT-6 Astra	gpt-6-astra	Codex native	xhigh	2	10
Claude Fable 5.1	claude-fable-5-1	Claude native (Claude Code)	max	2	10
GPT-5.6 Sol	gpt-5.6-sol	Codex native	xhigh	2	10
GPT-5.6 Terra	gpt-5.6-terra	Codex native	xhigh	2	10
Claude Sonnet 5	claude-sonnet-5	Claude native (Claude Code)	xhigh	2	10
DeepSeek V4 Pro	deepseek-v4-pro-0813	OpenRouter adapter	xhigh	2	10
GLM-5.3 Flash	z-ai/glm-5.3-flash	OpenRouter adapter	xhigh	2	10

Sandbox. Each run has its own workspace, isolated by Bubblewrap at the filesystem level and by a seccomp policy that blocks all network access except Magma’s license check. The environment provides Python 3.12 with SymPy and mpmath, GAP 4.15.1 with `ctbllib`, `atlasrep`, `tomllib`, and `mapclass`, Magma V2.29-9, and `ripgrep`; the GPT-6 Sol runs used Python 3.11.16 and Magma V2.26-5. Jobs run on an institutional HPC cluster, one task per job, with eight cores.

Limits. A shell command issued by the agent times out after 7,200 s; many runs, among them the Opus runs and ten GPT-5.6 runs, were configured with 1,800 s. A Magma call made by a checkpoint verifier is limited to 21,600 s, a braid-orbit enumeration to 2×10^6 classes, and a Nielsen enumeration to 2×10^6 candidate tuples. A verifier call that exceeds its limit returns `unsupported`, and the checkpoint is not credited.

Runs. The analysis covers 90 runs, two per task for each of the nine systems in Table 4. *Stored answers* are Magma’s polynomial database, stored number fields, and stored Belyi maps; they were blocked for Opus, GPT-6 Sol, Fable, and the second runs of GPT-5.6 Sol.

F GALOIS DETAILS

Figure 8 shows the control loop of Section 4.

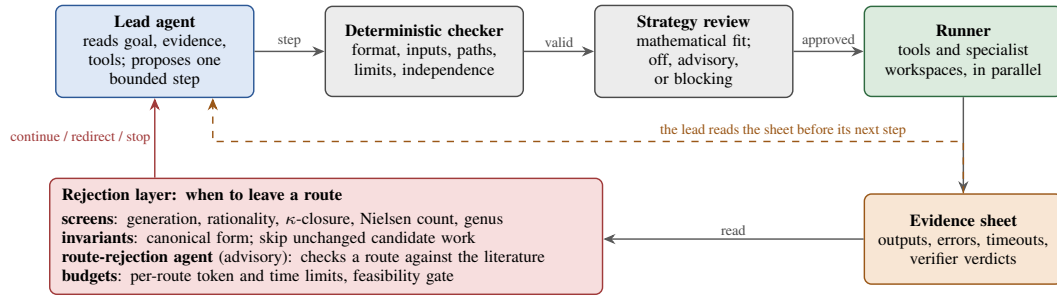


Figure 8: **The GALOIS control loop.** The lead proposes one resource-limited step. A deterministic checker validates it, an optional strategy reviewer checks its mathematical fit, and a runner executes it using registered tools or specialist workspaces. Outputs, timeouts, and errors go into an evidence sheet that the lead reads before its next step. The rejection layer uses cheap screens to admit and rank candidates, normalization invariants to avoid repeating unchanged work, and a route-rejection agent that checks routes against the literature. This agent only advises; rejecting a route requires an exact certificate. Budget rules may defer work without proving the route impossible.

Run configuration. The reference runs of Table 2 use Claude Opus 5.5 at medium effort, through the Claude CLI, as the lead agent. Strategy review is off, the route-rejection agent is on, and there is no spend cap. Each run starts from a fresh checkout that contains only the files needed at run time. Heavy tools run on one workstation with 16 cores and 110 GB of memory. The 36 skills encode methods of the constructive literature: enumeration of class vectors and rigidity screens, numerical uniformization, recognition, descent, and certification.

Scores of the reference runs. The reference runs produce no artifact graph, and their row in Table 3 is derived from the records of the control-loop steps. All ten runs reach a verified polynomial (Table 2), whose Galois group we recomputed in Magma after the runs; we therefore assign $C = 40$ and set S and R to 30, hence $P = 100$. K_r is computed from the output tokens of the control-loop steps, with steps in place of artifacts, and the entry for U_r counts screening and every step on the finished route as useful.

G ADDITIONAL RESULTS

This appendix gives the analyses behind Section 6, in the order of its claims, and then the scores by task and by run.

G.1 UNCERTAINTY AND SENSITIVITY

Table 5 gives 95% bootstrap intervals for the system means, with runs resampled and, alternatively, with tasks resampled. If a route counts as correct only when the literature supports it for the target

group itself, the mean progress scores are 76.2 for Opus, 62.1 for GPT-6 Sol, 59.4 for Astra, and 57.0 for Fable. With retrospective labels wherever they exist, the mean token-based K_r changes from 55.9 to 58.0 for Astra, from 92.7 to 97.3 for Fable (98.0 on its first batch, Table 8), and from 62.9 to 57.8 for GPT-5.6 Sol, and it is unchanged for the other systems at the reported precision. With any one task left out, Opus keeps the highest mean progress score; without M_{23} , GPT-6 Sol and Astra have 65.1 and 65.5.

Table 5: System means with 95% bootstrap intervals from 10,000 resamples. *Over runs*: percentile intervals from resampling runs with replacement. *Task-clustered*: percentile intervals from resampling the five tasks with replacement, keeping every run of each sampled task.

System	n	over runs			task-clustered	
		P	R	K_r (tokens)	P	K_r (tokens)
Claude Opus 5	10	81.0 [70, 92]	100.0 [100, 100]	72.8 [56, 87]	[66, 96]	[57, 89]
GPT-6 Sol	10	62.1 [48, 76]	80.0 [50, 100]	79.1 [67, 91]	[54, 72]	[64, 93]
GPT-6 Astra	10	59.4 [40, 78]	50.0 [20, 80]	55.9 [34, 77]	[46, 72]	[27, 82]
Claude Fable 5.1	10	62.4 [45, 80]	70.0 [40, 100]	92.7 [83, 99]	[45, 83]	[85, 99]
GPT-5.6 Sol	10	44.2 [27, 62]	50.0 [20, 80]	62.9 [48, 78]	[27, 58]	[47, 77]
GPT-5.6 Terra	10	21.9 [9, 36]	20.0 [0, 50]	38.2 [23, 53]	[9, 38]	[22, 54]
Claude Sonnet 5	10	17.5 [8, 29]	10.0 [0, 30]	30.6 [14, 48]	[7, 28]	[13, 48]
DeepSeek V4 Pro	10	7.8 [5, 11]	0.0 [0, 0]	66.6 [49, 83]	[6, 11]	[46, 84]
GLM-5.3 Flash	10	2.6 [0, 6]	0.0 [0, 0]	63.9 [51, 77]	[0, 5]	[49, 76]

G.2 CHOOSING A ROUTE AND GOING DEEPER ON IT

Section 6 compares choosing a route with going deeper on it over the 50 runs of five systems: Claude Opus 5, GPT-6 Astra, Claude Fable 5.1, GPT-6 Sol, and GPT-5.6 Sol. These five lead whether the systems are ranked by mean construction score or by mean route-identification score. Over their runs, $\rho(C, R) = +0.05$; over the 40 runs of the four systems of the current model generation, a grouping that uses no score, $\rho(C, R) = +0.02$ (Table 6). Over all 90 runs the correlation is $+0.29$, because pooling turns the gap between the five systems and the other four into a correlation. Among the 50 runs, 14 of the 16 with a verified polynomial earn R , against 21 of the 34 without; the difference is not significant at this sample size (Fisher’s exact test, $p = 0.10$).

The construction score has correlation $+0.08$ ($[-0.20, +0.35]$) with the rest of the score, $S + R$; intervals for correlations are 95% percentile bootstrap intervals over runs. Over the same 50 runs, its correlation of $+0.60$ with P is no larger than what the identity $P = S + R + C$ alone produces: when $S + R$ is permuted at random among the runs, the correlation is $+0.69$ on average over 20,000 permutations.

Table 7 gives the point contributions behind the comparison of Opus, GPT-6 Sol, and Astra in Section 6: Astra obtains 12 more construction points than GPT-6 Sol, but 5.7 fewer screening points and 9 fewer route-identification points.

Table 6: Spearman correlation between the construction score C and the route-identification score R , with 95% bootstrap intervals from 10,000 resamples of runs. *The five systems that make progress*: Claude Opus 5, GPT-6 Astra, Claude Fable 5.1, GPT-6 Sol, and GPT-5.6 Sol. *Current model generation*: Claude Opus 5, Claude Fable 5.1, GPT-6 Sol, and GPT-6 Astra. *POLY*: the number of runs with a verified polynomial. *R rate*: the fraction of runs that earn R . $\text{sd}(R)$: the standard deviation of R in points.

Group	Runs	POLY	R rate	$\rho(C, R)$	95% CI	$\text{sd}(R)$
The five systems that make progress	50	16	0.70	+0.05	$[-0.21, +0.30]$	13.9
Current model generation	40	15	0.75	+0.02	$[-0.25, +0.30]$	13.2
All systems pooled	90	16	0.42	+0.29	$[+0.07, +0.48]$	14.9

Table 7: **The construction score and the progress score rank Claude Opus 5, GPT-6 Sol, and GPT-6 Astra differently.** S , R , and C are mean contributions to P in points, with maxima 30, 30, and 40. The last column gives C as the percentage used in Table 3.

System	S	R	C	P	C (%)
Claude Opus 5	30.0	30.0	21.0	81.0	52.5
GPT-6 Sol	29.1	24.0	9.0	62.1	22.5
GPT-6 Astra	23.4	15.0	21.0	59.4	52.5

G.3 KNOWING WHEN TO STOP

Table 8 reports K_r and U_r by system in the three resource units, together with the number of routes researched per run, and Figure 9 gives the Spearman correlations between all pairs of metrics. Over the 50 runs of Appendix G.2, the correlation between K_r and C is $+0.01$, and the correlation between K_r and U_r is $+0.29$ ($[-0.02, +0.56]$). The within-task value of the former, $+0.25$ ($[-0.05, +0.51]$), is the correlation of the ranks after the mean rank of each task is subtracted. Restricting instead to the 38 of the 90 runs that earn R gives $\rho(C, K_r) = +0.03$ and $\rho(C, U_r) = +0.67$: how far a run goes on a route tracks its resource efficiency, not its K_r . Over the 50 runs, K_r has correlation $+0.43$ ($[+0.16, +0.65]$) with P .

Table 8: Stopping statistics by system, means over runs. *Routes/run*: the number of researched routes and the number of wrong routes among them, from the retrospective labels. K_r and U_r are given for output tokens, tool CPU time, and worker time. K_r uses the evaluator’s route labels where they are recorded and retrospective labels otherwise; for GPT-6 Astra and eight DeepSeek runs the evaluator’s value is recorded for output tokens only, and the CPU and worker columns use retrospective labels. The column *retro.* uses retrospective labels for every run, in output tokens. The first K_r column is the one in Table 3. Fable’s second batch has no retrospective labels: its route counts come from the evaluator’s record, and its entry under *retro.* averages the first batch only. *CPU h*: mean tool CPU hours per run. Means exclude undefined values, namely four GLM runs in every unit and one Sonnet run for CPU time.

System	n	Routes/run		K_r			K_r		U_r		
		all	wrong	tok.	CPU	worker	retro.	tok.	CPU	worker	CPU h
Claude Opus 5	10	10.5	9.3	72.8	53.0	66.0	72.8	57.7	60.2	56.3	2.2
GPT-6 Sol	10	20.5	19.3	79.1	61.1	75.6	79.1	31.4	29.7	30.7	0.7
GPT-6 Astra	10	3.8	2.9	55.9	54.4	59.4	58.0	60.3	62.8	56.8	0.4
Claude Fable 5.1	10	22.9	21.9	92.7	68.8	84.9	98.0	72.2	67.8	61.0	5.7
GPT-5.6 Sol	10	9.3	8.8	62.9	38.8	53.3	57.8	27.5	23.3	23.7	2.6
GPT-5.6 Terra	10	6.9	6.6	38.2	34.2	31.6	38.2	24.6	13.8	16.9	3.8
Claude Sonnet 5	10	3.9	3.9	30.6	8.2	20.3	30.6	26.9	28.5	26.3	2.4
DeepSeek V4 Pro	10	2.7	2.7	66.6	44.5	51.2	66.6	34.4	20.7	25.9	2.5
GLM-5.3 Flash	10	1.4	1.4	63.9	48.9	55.4	63.9	33.5	35.9	31.9	0.5

The M_{23} runs. Table 9 lists the 18 runs on M_{23} . Sixteen of them have the same construction score, the 5 points of checkpoint 2 (12.5 in the table), while their token-based K_r ranges from 4.7 to 98.2. The runs that identify the route complete steps 1–4 of Appendix C and parts of steps 5 and 6; none completes step 8. The first Astra run ($K_r = 4.7$) illustrates work that continues past an obstruction: artifact 4 records that inversion exchanges 23A and 23B, which obstructs the route (2A, 4A, 23A) over $\mathbb{Q}$, and artifacts 15–25 then build a cover on that route over a quartic field (Figure 2).

Route labels and R . Earning R and researching a route that is retrospectively labeled correct are different events; Table 10 lists the 15 runs in which they differ.

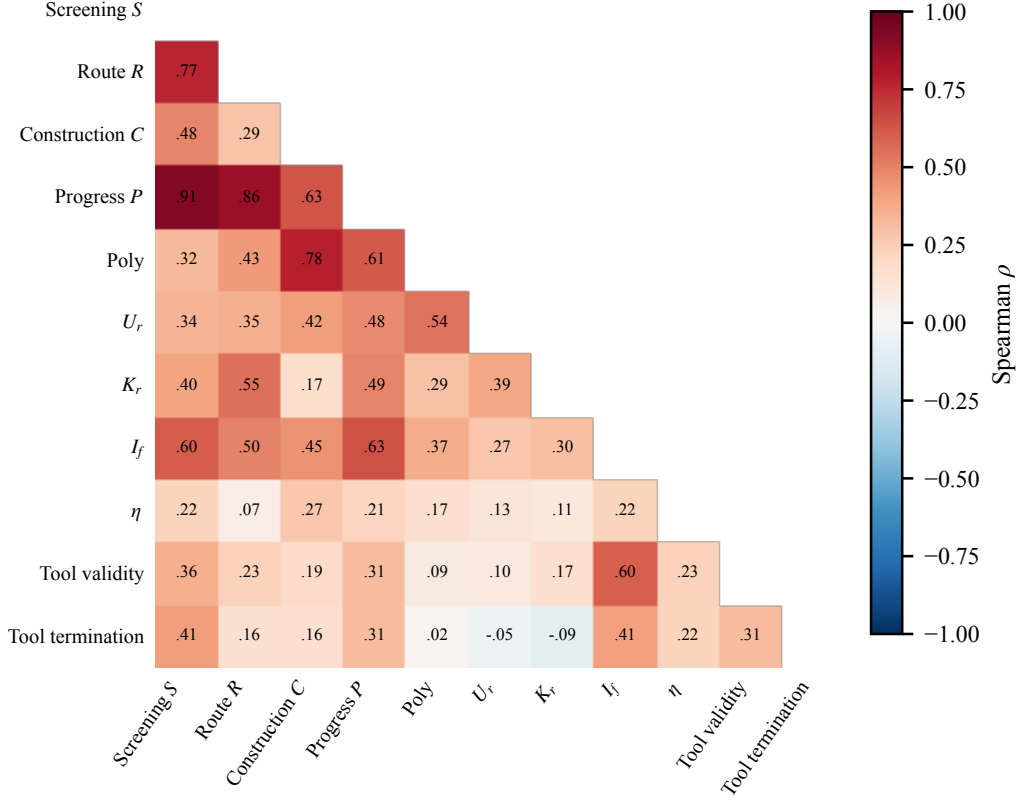


Figure 9: Spearman correlations between the IGP-Bench metrics over the 90 runs. Each pair uses the runs on which both metrics are defined: K_r is defined on 86 runs, I_f on 89, and η on 85.

Table 9: The 18 runs on M_{23} . *Routes* and *Wrong*: the number of researched routes and the number of wrong routes among them, from the retrospective labels (from the evaluator’s record for Fable’s run 2). K_r is given for output tokens and for tool CPU time, U_r for output tokens. $C = 12.5$ is 5 points out of 40, the value of checkpoint 2 of the M_{23} ladder (Appendix C). N/A: the run researches no route, so K_r is undefined.

System	Run	P	R	C	Routes	Wrong	K_r tok.	K_r CPU	U_r tok.	Tokens (k)	Hours
Claude Opus 5	run 1	65	100	12.5	1	1	17.6	9.3	22.9	451	5.6
	run 2	65	100	12.5	10	9	78.6	46.1	35.7	460	10.0
GPT-6 Sol	run 1	35	0	12.5	25	25	43.8	4.3	31.2	158	10.0
	run 2	65	100	12.5	22	21	81.1	8.7	33.0	154	10.0
GPT-6 Astra	run 1	35	0	12.5	8	8	4.7	0.9	3.5	500	7.4
	run 2	35	0	12.5	13	13	6.9	3.5	6.1	504	8.2
Claude Fable 5.1	run 1	35	0	12.5	107	106	95.7	3.7	27.1	310	10.0
	run 2	35	0	12.5	53	53	88.3	8.6	21.8	360	10.0
GPT-5.6 Sol	run 1	35	0	12.5	11	11	39.7	4.1	20.0	83	3.5
	run 2	35	0	12.5	5	5	24.4	1.5	14.5	232	9.8
GPT-5.6 Terra	run 1	35	0	12.5	7	7	13.4	0.1	50.7	92	3.9
	run 2	65	100	12.5	14	13	62.4	0.8	23.2	290	10.0
Claude Sonnet 5	run 1	24	0	12.5	5	5	77.7	63.3	48.3	451	4.4
	run 2	35	0	12.5	2	2	40.1	0.0	64.0	411	5.8
DeepSeek V4 Pro	run 1	20	0	12.5	3	3	83.1	12.1	2.1	242	1.9
	run 2	10	0	12.5	11	11	98.2	96.4	15.5	211	2.6
GLM-5.3 Flash	run 1	0	0	0.0	0	0	N/A	N/A	0.0	509	2.4
	run 2	0	0	0.0	2	2	48.1	97.5	0.0	446	5.0

Table 10: The 15 runs in which the route-identification score and the retrospective route labels disagree: nine research a route labeled correct without earning R , and six earn R without researching one. *Correct* and *Wrong*: the numbers of researched routes with each retrospective label. K_r is the value of Table 14, which uses the evaluator’s labels where they are recorded. Fable’s second batch has no retrospective labels and is not compared.

System	Task	Run	R	Correct	Wrong	K_r tok.
GPT-6 Sol	M12	run 2	0	1	28	83.1
GPT-6 Astra	M11	run 1	0	1	1	28.6
GPT-6 Astra	M12	run 2	0	1	3	63.2
GPT-6 Astra	M22	run 2	0	1	1	47.9
Claude Fable 5.1	M11	run 1	0	2	0	54.1
Claude Fable 5.1	M23	run 1	0	1	106	95.7
GPT-5.6 Sol	M12	run 1	0	1	18	78.3
GPT-5.6 Sol	M12	run 2	0	1	7	37.5
GPT-5.6 Terra	M11	run 2	0	1	3	67.0
Claude Opus 5	M23	run 1	100	0	1	17.6
GPT-6 Sol	M22	run 1	100	0	17	60.5
GPT-6 Sol	M22	run 2	100	0	11	49.5
GPT-5.6 Sol	M22	run 2	100	0	6	93.0
GPT-5.6 Sol	M24	run 2	100	0	6	40.9
Claude Sonnet 5	M22	run 1	100	0	8	69.9

G.4 TASKS UNDER THE OPEN-CASE PROFILE

Table 11 reports one Claude Opus 5 run on each of ten tasks, scored under the open-case profile (Appendix D.4). The ten tasks were selected by a Magma screen of class triples that excludes groups with a rigid triple of rational classes; five of them are Tier 2 and five are Tier 3 under the classification used here, which retains the experimental Tier 3 label for $\text{PSL}_2(16)$. Each run had maximum reasoning effort, 500,000 output tokens, eight CPUs, and at most 2 h 48 min of agent time. Six runs resolved a candidate passport by a proved obstruction, each within the first hour and a half. All ten reached the braid-orbit checkpoint, and none claimed a cover.

Table 11: **Ten tasks under the open-case profile.** One Claude Opus 5 run per task at maximum effort (Appendix G.4). S , R , C , and P are raw points, with maxima 30, 30, 40, and 100. *level*: the deepest verified checkpoint. *cand.*: the number of candidate passports; *res.*: the number that the run resolved. *resolution*: the passport, its genus, and the obstruction (N4 or N7, Appendix D.4); for $\text{PSL}_3(7)$, two of the four resolved passports are listed. *tok.*: output tokens. Class names follow the GAP character-table library.

Group	$g_{\max}$	cand.	res.	S	R	C	P	level	resolution (passport, genus, obstruction)	tok.
Sz(8)	4	1	1	30	30	0	60	S2	(2a, 5a, 5a), 4, N7	371k
PSU ₃ (4)	2	1	1	30	30	0	60	S2	(3a, 3a, 4a), 2, N7	348k
PSU ₃ (5)	2	16	2	30	30	0	60	S2	(2a, 5a, 6a), 0, N4. (2a, 4a, 10a), 0, N7	373k
PSL ₃ (7)	2	13	4	30	30	0	60	S2	(2a, 4a, 7b), 0, N7. (2a, 4a, 14a), 0, N7	402k
PSL ₄ (3)	2	211	1	30	30	0	60	S2	(3a, 5a, 8a), 2, N7	363k
PSL ₂ (16)	2	2	2	30	30	0	60	S2	(2a, 5a, 5b), 0, N4. (3a, 5a, 5b), 1, N4	288k
PSL ₂ (125)	2	1	0	23	0	0	23	S2	none. N7 only on (7a, 7b, 7c) of genus 37	344k
PSU ₃ (8)	2	4	0	23	0	0	23	S2	none. N7 only on passports of genus 22 and 64	467k
Sz(32)	2	0	0	23	0	0	23	S2	none. No three-point candidate of genus ≤ 12	290k
PSL ₂ (27)	9	1	0	22	0	0	22	S2	none	441k
mean				27.1	18.0	0.0	45.1			

G.5 SCORES BY TASK AND BY RUN

Table 12 gives the mean scores of each system on each task, Table 13 records how the runs ended, and Table 14 lists the 90 runs behind the means of Table 3.

Table 12: Verified progress by system and task, means over the two runs. S , R , and C are percentages of their maxima; P is out of 100.

System	Task	S	R	C	P	POLY	n
Claude Opus 5	M11	100.0	100.0	50.0	80.0	50.0	2
Claude Opus 5	M12	100.0	100.0	100.0	100.0	100.0	2
Claude Opus 5	M22	100.0	100.0	100.0	100.0	100.0	2
Claude Opus 5	M23	100.0	100.0	12.5	65.0	0.0	2
Claude Opus 5	M24	100.0	100.0	0.0	60.0	0.0	2
GPT-6 Sol	M11	100.0	100.0	0.0	60.0	0.0	2
GPT-6 Sol	M12	85.0	50.0	50.0	60.5	50.0	2
GPT-6 Sol	M22	100.0	100.0	50.0	80.0	50.0	2
GPT-6 Sol	M23	100.0	50.0	12.5	50.0	0.0	2
GPT-6 Sol	M24	100.0	100.0	0.0	60.0	0.0	2
GPT-6 Astra	M11	56.7	50.0	50.0	52.0	50.0	2
GPT-6 Astra	M12	60.0	50.0	100.0	73.0	100.0	2
GPT-6 Astra	M22	73.3	50.0	100.0	77.0	100.0	2
GPT-6 Astra	M23	100.0	0.0	12.5	35.0	0.0	2
GPT-6 Astra	M24	100.0	100.0	0.0	60.0	0.0	2
Claude Fable 5.1	M11	73.3	50.0	50.0	57.0	50.0	2
Claude Fable 5.1	M12	100.0	100.0	0.0	60.0	0.0	2
Claude Fable 5.1	M22	100.0	100.0	100.0	100.0	100.0	2
Claude Fable 5.1	M23	100.0	0.0	12.5	35.0	0.0	2
Claude Fable 5.1	M24	100.0	100.0	0.0	60.0	0.0	2
GPT-5.6 Sol	M11	100.0	100.0	0.0	60.0	0.0	2
GPT-5.6 Sol	M12	46.7	0.0	0.0	14.0	0.0	2
GPT-5.6 Sol	M22	56.7	50.0	50.0	52.0	50.0	2
GPT-5.6 Sol	M23	100.0	0.0	12.5	35.0	0.0	2
GPT-5.6 Sol	M24	100.0	100.0	0.0	60.0	0.0	2
GPT-5.6 Terra	M11	36.7	0.0	0.0	11.0	0.0	2
GPT-5.6 Terra	M12	56.7	50.0	0.0	32.0	0.0	2
GPT-5.6 Terra	M22	23.3	0.0	0.0	7.0	0.0	2
GPT-5.6 Terra	M23	100.0	50.0	12.5	50.0	0.0	2
GPT-5.6 Terra	M24	31.7	0.0	0.0	9.5	0.0	2
Claude Sonnet 5	M11	26.7	0.0	0.0	8.0	0.0	2
Claude Sonnet 5	M12	20.0	0.0	0.0	6.0	0.0	2
Claude Sonnet 5	M22	66.7	50.0	0.0	35.0	0.0	2
Claude Sonnet 5	M23	81.7	0.0	12.5	29.5	0.0	2
Claude Sonnet 5	M24	30.0	0.0	0.0	9.0	0.0	2
DeepSeek V4 Pro	M11	20.0	0.0	0.0	6.0	0.0	2
DeepSeek V4 Pro	M12	20.0	0.0	0.0	6.0	0.0	2
DeepSeek V4 Pro	M22	20.0	0.0	0.0	6.0	0.0	2
DeepSeek V4 Pro	M23	33.3	0.0	12.5	15.0	0.0	2
DeepSeek V4 Pro	M24	20.0	0.0	0.0	6.0	0.0	2
GLM-5.3 Flash	M11	16.7	0.0	0.0	5.0	0.0	2
GLM-5.3 Flash	M12	6.7	0.0	0.0	2.0	0.0	2
GLM-5.3 Flash	M22	0.0	0.0	0.0	0.0	0.0	2
GLM-5.3 Flash	M23	0.0	0.0	0.0	0.0	0.0	2
GLM-5.3 Flash	M24	20.0	0.0	0.0	6.0	0.0	2

Table 13: How the runs ended: the recorded reason, and the number of runs whose counters reached 99% of the wall-time budget or of the token budget.

System	Recorded termination				Derived from counters	
	Completed	Wall limit	Token limit	Other	Wall $\geq$ 99%	Tokens $\geq$ 99%
Claude Opus 5	7	2	1	0	3	1
GPT-6 Sol	4	3	3	0	3	3
GPT-6 Astra	7	1	2	0	1	2
Claude Fable 5.1	5	4	1	0	4	1
GPT-5.6 Sol	9	1	0	0	1	0
GPT-5.6 Terra	4	6	0	0	6	0
Claude Sonnet 5	10	0	0	0	0	1
DeepSeek V4 Pro	4	0	6	0	0	6
GLM-5.3 Flash	10	0	0	0	1	7

Table 14: Scores of all 90 runs. The columns are those of Table 3; S , R , and C are percentages of their maxima, and POLY is 100 for a run with a verified polynomial. Every run had 10 hours and 500,000 output tokens; *run 1* and *run 2* are the two runs of a system on a task. N/A: the metric is undefined for the run (Appendix D).

Task	Run	Verified progress (%)					Allocation		Protocol		Tool use (%)	
		<i>S</i>	<i>R</i>	<i>C</i>	<i>P</i>	POLY	<i>U_r</i>	<i>K_r</i>	<i>I_f</i>	η	Valid	Term.
Claude Opus 5 (10 runs)												
M11	run 1	100.0	100.0	0.0	60.0	0.0	21.0	89.8	98.5	100.0	100.0	96.1
	run 2	100.0	100.0	100.0	100.0	100.0	58.8	89.8	100.0	100.0	100.0	95.6
M12	run 1	100.0	100.0	100.0	100.0	100.0	94.8	100.0	86.4	100.0	99.4	87.4
	run 2	100.0	100.0	100.0	100.0	100.0	95.5	95.1	95.0	100.0	100.0	83.8
M22	run 1	100.0	100.0	100.0	100.0	100.0	95.1	40.7	100.0	100.0	100.0	81.2
	run 2	100.0	100.0	100.0	100.0	100.0	96.2	76.7	100.0	90.0	99.3	88.2
M23	run 1	100.0	100.0	12.5	65.0	0.0	22.9	17.6	96.9	100.0	100.0	91.3
	run 2	100.0	100.0	12.5	65.0	0.0	35.7	78.6	97.7	100.0	100.0	94.9
M24	run 1	100.0	100.0	0.0	60.0	0.0	29.0	78.2	97.9	100.0	100.0	97.1
	run 2	100.0	100.0	0.0	60.0	0.0	28.2	61.4	96.7	100.0	99.6	96.1
GPT-6 Sol (10 runs)												
M11	run 1	100.0	100.0	0.0	60.0	0.0	2.4	92.8	99.1	100.0	100.0	98.8
	run 2	100.0	100.0	0.0	60.0	0.0	3.8	90.4	98.9	100.0	100.0	98.6
M12	run 1	100.0	100.0	100.0	100.0	100.0	64.5	95.9	99.2	100.0	100.0	98.3
	run 2	70.0	0.0	0.0	21.0	0.0	19.0	83.1	100.0	100.0	100.0	97.9
M22	run 1	100.0	100.0	100.0	100.0	100.0	83.3	60.5	99.0	100.0	100.0	97.6
	run 2	100.0	100.0	0.0	60.0	0.0	64.5	49.5	98.8	88.9	100.0	99.5
M23	run 1	100.0	0.0	12.5	35.0	0.0	31.2	43.8	100.0	100.0	100.0	99.8
	run 2	100.0	100.0	12.5	65.0	0.0	33.0	81.1	100.0	97.6	100.0	99.5
M24	run 1	100.0	100.0	0.0	60.0	0.0	3.0	96.9	98.4	98.2	100.0	99.1
	run 2	100.0	100.0	0.0	60.0	0.0	9.6	97.1	99.0	98.9	100.0	98.3
GPT-6 Astra (10 runs)												
M11	run 1	13.3	0.0	0.0	4.0	0.0	99.3	28.6	100.0	100.0	100.0	95.7
	run 2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	97.4
M12	run 1	100.0	100.0	100.0	100.0	100.0	65.3	45.8	95.0	100.0	99.0	97.1
	run 2	20.0	0.0	100.0	46.0	100.0	41.5	63.2	95.0	100.0	99.0	99.0
M22	run 1	100.0	100.0	100.0	100.0	100.0	100.0	62.9	100.0	100.0	100.0	99.2
	run 2	46.7	0.0	100.0	54.0	100.0	77.4	47.9	100.0	100.0	100.0	97.3
M23	run 1	100.0	0.0	12.5	35.0	0.0	3.5	4.7	97.2	100.0	100.0	98.2
	run 2	100.0	0.0	12.5	35.0	0.0	6.1	6.9	97.2	100.0	99.8	98.4
M24	run 1	100.0	100.0	0.0	60.0	0.0	60.8	99.1	100.0	100.0	100.0	92.7
	run 2	100.0	100.0	0.0	60.0	0.0	49.0	100.0	97.6	100.0	99.8	91.8
Claude Table 5.1 (10 runs)												
M11	run 1	46.7	0.0	0.0	14.0	0.0	100.0	54.1	93.8	100.0	100.0	94.8
	run 2	100.0	100.0	100.0	100.0	100.0	91.9	100.0	100.0	100.0	100.0	89.9
M12	run 1	100.0	100.0	0.0	60.0	0.0	81.2	100.0	70.0	100.0	100.0	85.9
	run 2	100.0	100.0	0.0	60.0	0.0	100.0	100.0	100.0	75.0	100.0	79.6
M22	run 1	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	89.7
	run 2	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	85.0
M23	run 1	100.0	0.0	12.5	35.0	0.0	27.1	95.7	97.4	100.0	100.0	97.7
	run 2	100.0	0.0	12.5	35.0	0.0	21.8	88.3	97.4	100.0	100.0	98.7
M24	run 1	100.0	100.0	0.0	60.0	0.0	27.0	94.1	97.1	100.0	100.0	93.9
	run 2	100.0	100.0	0.0	60.0	0.0	72.6	94.7	100.0	100.0	100.0	96.1
GPT-5.6 Sol (10 runs)												
M11	run 1	100.0	100.0	0.0	60.0	0.0	46.6	83.4	97.9	100.0	100.0	93.8
	run 2	100.0	100.0	0.0	60.0	0.0	20.9	83.5	97.1	100.0	99.3	94.4
M12	run 1	43.3	0.0	0.0	13.0	0.0	26.7	78.3	99.0	100.0	100.0	96.6
	run 2	50.0	0.0	0.0	15.0	0.0	27.1	37.5	98.3	100.0	99.8	89.7
M22	run 1	13.3	0.0	0.0	4.0	0.0	5.8	58.4	98.4	100.0	100.0	92.8
	run 2	100.0	100.0	100.0	100.0	100.0	90.5	93.0	100.0	100.0	100.0	93.9

continued on next page

Table 14 continued

Task	Run	Verified progress (%)					Allocation		Protocol		Tool use (%)	
		<i>S</i>	<i>R</i>	<i>C</i>	<i>P</i>	POLY	<i>U_r</i>	<i>K_r</i>	<i>I_f</i>	η	Valid	Term.
M23	run 1	100.0	0.0	12.5	35.0	0.0	20.0	39.7	97.7	100.0	100.0	96.3
	run 2	100.0	0.0	12.5	35.0	0.0	14.5	24.4	97.5	100.0	99.8	95.3
M24	run 1	100.0	100.0	0.0	60.0	0.0	8.3	89.6	100.0	100.0	100.0	95.5
	run 2	100.0	100.0	0.0	60.0	0.0	14.6	40.9	97.9	80.0	100.0	93.7
GPT-5.6 Terra (10 runs)												
M11	run 1	33.3	0.0	0.0	10.0	0.0	16.5	45.2	96.4	100.0	100.0	98.4
	run 2	40.0	0.0	0.0	12.0	0.0	30.9	67.0	92.9	100.0	100.0	93.7
M12	run 1	13.3	0.0	0.0	4.0	0.0	3.2	54.8	98.1	100.0	100.0	93.5
	run 2	100.0	100.0	0.0	60.0	0.0	82.7	62.8	92.9	100.0	99.1	95.3
M22	run 1	13.3	0.0	0.0	4.0	0.0	4.7	5.1	94.7	100.0	100.0	90.7
	run 2	33.3	0.0	0.0	10.0	0.0	7.6	8.3	97.4	97.1	100.0	93.8
M23	run 1	100.0	0.0	12.5	35.0	0.0	50.7	13.4	100.0	100.0	100.0	91.5
	run 2	100.0	100.0	12.5	65.0	0.0	23.2	62.4	98.8	100.0	99.8	94.5
M24	run 1	36.7	0.0	0.0	11.0	0.0	17.1	51.0	95.7	100.0	99.8	96.7
	run 2	26.7	0.0	0.0	8.0	0.0	9.1	11.9	96.2	100.0	100.0	91.7
Claude Sonnet 5 (10 runs)												
M11	run 1	33.3	0.0	0.0	10.0	0.0	1.0	1.0	88.9	80.0	99.6	93.6
	run 2	20.0	0.0	0.0	6.0	0.0	0.3	0.0	93.8	100.0	100.0	95.5
M12	run 1	0.0	0.0	0.0	0.0	0.0	39.7	40.3	90.9	100.0	100.0	96.1
	run 2	40.0	0.0	0.0	12.0	0.0	16.8	0.0	94.4	92.9	99.6	92.9
M22	run 1	100.0	100.0	0.0	60.0	0.0	0.7	69.9	94.7	100.0	99.5	96.4
	run 2	33.3	0.0	0.0	10.0	0.0	40.1	19.9	95.7	100.0	100.0	96.4
M23	run 1	63.3	0.0	12.5	24.0	0.0	48.3	77.7	97.4	100.0	99.6	96.4
	run 2	100.0	0.0	12.5	35.0	0.0	64.0	40.1	93.8	100.0	100.0	95.8
M24	run 1	20.0	0.0	0.0	6.0	0.0	1.1	11.3	94.7	100.0	99.8	95.2
	run 2	40.0	0.0	0.0	12.0	0.0	56.5	46.0	94.4	100.0	99.7	97.0
DeepSeek V4 Pro (10 runs)												
M11	run 1	20.0	0.0	0.0	6.0	0.0	16.9	44.5	88.9	100.0	100.0	91.9
	run 2	20.0	0.0	0.0	6.0	0.0	81.1	82.6	80.0	66.7	100.0	88.4
M12	run 1	20.0	0.0	0.0	6.0	0.0	0.3	23.6	80.0	66.7	100.0	89.0
	run 2	20.0	0.0	0.0	6.0	0.0	16.5	26.4	90.0	75.0	98.5	94.8
M22	run 1	0.0	0.0	0.0	0.0	0.0	18.4	81.5	75.0	75.0	99.4	89.6
	run 2	40.0	0.0	0.0	12.0	0.0	95.3	87.7	87.5	100.0	99.4	90.5
M23	run 1	50.0	0.0	12.5	20.0	0.0	2.1	83.1	95.0	100.0	100.0	92.5
	run 2	16.7	0.0	12.5	10.0	0.0	15.5	98.2	100.0	100.0	100.0	76.5
M24	run 1	20.0	0.0	0.0	6.0	0.0	58.0	99.3	83.3	100.0	100.0	87.7
	run 2	20.0	0.0	0.0	6.0	0.0	40.0	38.8	85.7	100.0	98.4	85.2
GLM-5.3 Flash (10 runs)												
M11	run 1	0.0	0.0	0.0	0.0	0.0	83.2	81.8	75.0	100.0	98.6	88.9
	run 2	33.3	0.0	0.0	10.0	0.0	9.6	64.8	62.5	100.0	97.0	78.1
M12	run 1	0.0	0.0	0.0	0.0	0.0	0.0	N/A	0.0	N/A	96.6	87.8
	run 2	13.3	0.0	0.0	4.0	0.0	63.7	56.2	66.7	66.7	98.5	84.4
M22	run 1	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A	100.0	75.4
	run 2	0.0	0.0	0.0	0.0	0.0	99.9	45.2	61.1	N/A	98.3	90.3
M23	run 1	0.0	0.0	0.0	0.0	0.0	0.0	N/A	33.3	N/A	94.9	86.5
	run 2	0.0	0.0	0.0	0.0	0.0	0.0	48.1	46.3	33.3	92.8	79.2
M24	run 1	40.0	0.0	0.0	12.0	0.0	78.1	87.1	72.2	75.0	97.3	78.7
	run 2	0.0	0.0	0.0	0.0	0.0	0.0	N/A	61.1	N/A	98.4	70.4