Identical Scores, Different Weak Spots: Cost Spans 4× at Equal Accuracy

An evaluation of Claude Fable 5 (low and medium), Claude Opus 4.8 (high), and GPT-5.5 (high) on fifteen decontaminated evals from production open-source repositories

VulcanBench (github.com/morganlinton/VulcanBench)

Abstract

We evaluate four frontier configurations — Claude Fable 5 at low and medium reasoning effort, Claude Opus 4.8 at high effort, and GPT-5.5 at high effort — on VulcanBench Evals 6.2.26, a fifteen-eval suite built exclusively from real merged pull requests in seven production open-source repositories (flask, sqlglot, redis-py, anyio, pytest, urllib3, poetry, packaging, chi, click, more-itertools; Python and Go), each merged after the models' training cutoffs and graded by deterministic hidden tests in a network-isolated Docker sandbox. All four configurations score 14 of 15 — but each misses a different task. GPT-5.5 is the first evaluated model to solve the multi-site Flask teardown redesign that stops every Anthropic model, yet it is also the only configuration to fail an SQL dialect transpile the others pass easily. At identical accuracy, total cost spans 4× ($5.00 to $20.43, with each provider's prompt-cache discount folded in) and token usage spans 7×. Opus 4.8 is the value leader. On ticket-sized engineering work, correctness no longer separates these models; economics and failure geography do.

1. Introduction

Report No. 3 showed three models landing the same score with the same failure set. This report sharpens the question with a suite composed so that measured difficulty spans routine to unsolved, and with a cross-vendor configuration set. The outcome is stronger than convergence: four configurations tie at 14/15 while disagreeing about *which* task is impossible. The suite now separates frontier models not by how many tasks they solve but by what kind of task they miss — and by what they spend.

2. Method

2.1 Grading and sandbox

Each eval is a real merged pull request: the repository is sliced at the base commit, the agent receives a terse issue and the working tree, and its patch is graded by hidden tests validated to fail before and pass after the human fix. Runs execute in a network-isolated Docker sandbox with identical tooling and step budgets. One run per task per configuration: $4 \times 15 = 60$ runs, zero refusals.

2.2 Cost correction

Two corrections make this run trustworthy relative to an earlier draft. First, the harness folds each provider's automatic prompt-cache discount into cost; a naive token count overstates GPT-5.5's spend by roughly 6× (its true total is $20.43). Second, two tasks that a safety classifier had intermittently refused were replaced with equally-hard, reliably-attempted tasks, giving a clean fifteen for every configuration.

Table 1. The fifteen evals, ranked by measured difficulty (how many configurations solved each, and at what effort).

#	Eval	Source (repository & PR)	Type	Lang.	Difficulty
1	`flask-teardown-robust`	pallets/flask #5928	bug fix	Python	very hard
2	`sqlglot-iso8601-nanos`	tobymao/sqlglot #7808	bug fix	Python	hard
3	`redis-explicit-none-metadata`	redis/redis-py #4081	feature	Python	hard
4	`sqlglot-unpivot-cte-star`	tobymao/sqlglot #7550	bug fix	Python	hard
5	`anyio-max-bytes-guard`	agronholm/anyio #1191	bug fix	Python	hard
6	`sqlglot-qualify-lateral-star`	tobymao/sqlglot #7802	bug fix	Python	hard
7	`pytest-approx-public-type`	pytest-dev/pytest #14647	feature	Python	medium
8	`urllib3-reject-invalid-host`	urllib3/urllib3 #5095	bug fix	Python	medium
9	`poetry-extras-duplicate-conflict`	python-poetry/poetry #10943	bug fix	Python	medium
10	`packaging-range-prerelease-policy`	pypa/packaging #1295	bug fix	Python	medium
11	`sqlglot-parser-tuple-subquery`	tobymao/sqlglot #7807	bug fix	Python	medium
12	`chi-readfrom-tee-doublecount`	go-chi/chi #1085	bug fix	Go	easy
13	`click-version-distribution-name`	pallets/click #3582	bug fix	Python	easy
14	`packaging-empty-name`	pypa/packaging #1305	bug fix	Python	easy
15	`more-itertools-interleave-empty`	more-itertools #1193	bug fix	Python	easy

Table 2. Aggregate results, one run per task, deterministic hidden-test grading. Costs corrected for provider prompt-cache discounts.

Model (effort)	Score	Cost	Time	Steps	Tokens
Claude Fable 5 (low)	14/15	$5.99	27 min	66	0.40 M
Claude Fable 5 (medium)	14/15	$11.10	44 min	84	0.64 M
Claude Opus 4.8 (high)	14/15	**$5.00**	27 min	96	0.71 M
GPT-5.5 (high)	14/15	$20.43	71 min	195	2.97 M

Figure 1. Resource consumption at equal (14/15) accuracy. Within each panel, bars are Fable 5 low, Fable 5 medium, Opus 4.8 high, GPT-5.5 high.

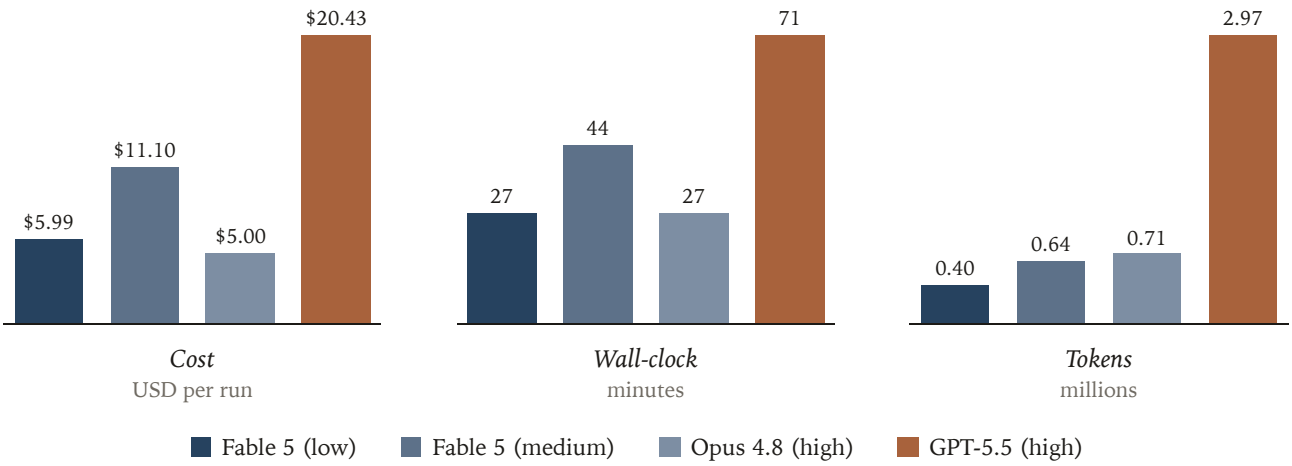

Table 3. The two discriminating evals. The remaining thirteen are solved by all four configurations at varying cost.

Eval	Fable 5 (low)	Fable 5 (med.)	Opus 4.8	GPT-5.5
`flask-teardown-robust`	fail	fail	fail	pass
`sqlglot-iso8601-nanos`	pass	pass	pass	fail

4. Findings

4.1 The single misses differ

Fable 5 (at both efforts) and Opus 4.8 fail only `flask-teardown-robust`, the multi-site behavioral redesign that has stopped every Anthropic model evaluated to date. GPT-5.5 is the first model to solve it — and the only one to fail `sqlglot-iso8601-nanos`, an SQL dialect transpile the others pass easily. The suite now separates these models by what kind of task they miss, not how many.

4.2 Opus 4.8 is the value leader

The same 14/15 as everything else, at the lowest cost ($5.00), the fewest tokens among comparable scores, and no reliability issues. For ticket-sized work at this accuracy level, it is the default choice on economics.

4.3 More effort is not always better

Fable 5 at medium effort costs 2× Fable 5 at low effort ($11.10 versus $5.99) for an identical score — the extra thinking budget bought nothing on this suite. GPT-5.5 spent 4× Opus 4.8 and used 4× the tokens to buy exactly one extra capability, which it offset by losing another. Effort and spend must be justified per-workload, not assumed.

5. Discussion

A four-way tie at 14/15 with disjoint failure sets is a more informative outcome than any ranking: the frontier has converged on breadth while still differing in reach, so each vendor's hardest-case behavior is now the distinguishing feature. Selection guidance follows directly. If the workload resembles the thirteen common tasks, choose on economics (Opus 4.8, then Fable 5 low). If it contains multi-site redesigns like the Flask teardown, GPT-5.5 currently holds the only demonstrated solve — at a 4× premium and with its own blind spot in dialect-heavy SQL work. For benchmark design, the result validates difficulty-spanned composition: only the two hardest evals separate the models, but without them the suite would report a meaningless four-way 100%.

6. Limitations

One run per task per configuration bounds cost, not variance; single-run cells should be read as point estimates, though the two discriminating results match repeated observations in earlier reports. Costs reflect each provider's prompt-cache discount at July 2026 list prices.

7. Reproducibility

Every eval, hidden test, gold patch, per-task Docker image, and the grading harness are open source; each run writes a full trace and replayable transcript. Reproducing any configuration costs $5–21: `vulcanbench run --suite evals-6.2.26 --model <model>`.