Kimi K3 vs Grok 4.5, Claude Fable 5 & GPT-5.6 Sol

Moonshot's 2.8T open-weight flagship joins the Report 07 field: twenty-three real merged post-cutoff PRs across Python, Rust, TypeScript, JavaScript, and Go, graded by deterministic hidden tests.

Abstract

Grok 4.5 leads all eleven configurations at 21/23 (91.3%) with the lowest cost per solved task ($0.32). Kimi K3 debuts at 17/23 (74%) in its standard configuration and 20/23 (87%) under an extended-budget ablation — level with the best Fable 5 and Sol configurations ($1.37 vs. $0.80–$1.04 per solved task). The two hardest v3 tasks remain unsolved by every model. Kimi benchmark spend $33.89 under a $40 cap.

Table 1. Results (Grok/Fable/Sol reproduced from Report 07)

Model (effort)	Score	pass@1	Cost	Tok/task	Time/task	$/solved
Grok 4.5 (medium)	**21/23**	**91.3%**	**$6.67**	**106 K**	**3.9 min**	**$0.32**
Grok 4.5 (high)	21/23	91.3%	$8.76	141 K	4.9 min	$0.42
Claude Fable 5 (low)*	20/23	87.0%	$12.18	28 K	3.4 min	$0.61
GPT-5.6 Sol (high)	20/23	87.0%	$15.90	85 K	4.2 min	$0.80
Claude Fable 5 (high)*	20/23	87.0%	$20.82	44 K	4.3 min	$1.04
Kimi K3 (max, 2-h budget)‡	20/23	87.0%	$27.43	216 K	28.3 min	$1.37
Grok 4.5 (low)	19/23	82.6%	$3.39	57 K	2.5 min	$0.18
GPT-5.6 Sol (medium)	19/23	82.6%	$8.83	50 K	2.8 min	$0.46
Claude Fable 5 (medium)*	19/23	82.6%	$16.32	36 K	3.7 min	$0.86
GPT-5.6 Sol (low)	18/23	78.3%	$3.85	23 K	1.5 min	$0.21
Kimi K3 (max, 30-min budget)	17/23	73.9%	$16.84	134 K	18.1 min	$0.99

*‡ 2-hour row: the five tasks that did not finish within the 30-minute budget were re-run with exact budgets of 2 h / 400 steps / $4 per run (runs marked budget_override); the other 18 runs unchanged. It is an ablation, not a leaderboard config — every other model ran under the standard budget. * Fable 5 ran with Opus 4.8 refusal fallback (see Report 07). Tokens are billed tokens; time is sandbox wall-clock.*

Findings

1. Grok 4.5 wins on accuracy and economics.
21/23 (91.3%) at $0.32 per solved task — no other configuration matches either number. Its medium and high efforts tie, keeping medium the efficiency frontier of the board.

2. Kimi K3 debuts mid-pack and closes with budget.
17/23 in its standard configuration; the extended-budget ablation converts aiohttp, NetworkX-Leiden, and jiff-strftime for +$10.59, reaching 20/23 — matching the best Fable 5 and Sol configurations.

3. The v3 difficulty ceiling held.
PennyLane Trotter fragmentation (0.2 partial) and SQLGlot internal-name canonicalization (0.17 partial) remain unsolved by every configuration ever run — 0-for-27 in Report 07, and K3's extended attempts didn't crack them either.

Extended-budget ablation

Task	30-min budget	2-h budget	Time	Cost
aiohttp-upgrade-deferred	x did not finish	ok solved	64 min	$2.66
networkx-leiden-communities	x did not finish	ok solved	107 min	$4.35
jiff-strftime-negpad	x did not finish	ok solved	21 min	$1.14
pennylane-trotter-fragmented	x did not finish	partial (0.2), $4 cap	100 min	$4.78
sqlglot-canonicalize-internal-names	x did not finish	partial (0.17), $4 cap	94 min	$4.12
sqlglot-iso8601-nanos	partial (0.7)	— not re-run	—	—

The two remaining tasks ended at the $4 per-run cost cap with partial progress.

Method & reproducibility

Same 23 v3 tasks as Report 07 (real merged post-cutoff PRs: Python 9, Rust 4, TypeScript 4, JavaScript 3, Go 3). Network-isolated Docker sandboxes; deterministic hidden-test grading; one attempt per task per configuration. Every task pre-validated: gold patch = 1.0, unpatched = 0.0, deterministic over 3 runs. kimi-k3 via the Moonshot API ($3/$15 per million tokens, cache-hit input billed here at the cache-miss rate), run in its single available API configuration. The extended-budget ablation runs are stamped budget_override so they cannot be mixed into standard comparisons. Reproduce: vulcanbench run --suite v3 --model kimi:kimi-k3 --effort extra-high — the ablation adds --tasks-root tasks/v3 --timeout 7200 --max-steps 400 --override-budgets --max-run-cost 4 per task.