Elis Token Saver: Cost, Accuracy, and Latency Evidence

Eight matched cases across two host model families

Generated: 2026-08-13 03:40 UTC

Elis Token Saver: Cost, Accuracy, and Latency Evidence

Version 0.1 - August 13, 2026
Status: Early internal study; eight matched cases across two host families

Abstract

Elis Token Saver privately routes intent, plans a task chain, and recommends the least expensive capable model tier while the host executes one task at a time. Across eight matched tasks on ChatGPT-family and Claude-family hosts, the Elis arm cost **56.7% less in aggregate** ($0.1410 versus $0.3258). Token volume did not move uniformly: it increased 40.7% in the ChatGPT-family run and decreased 28.9% in the Claude-family run. This distinction matters because model price, not token count alone, determines cost.

Accuracy was preserved at the resolution measured in the stronger Claude run: the plain arm passed 15 of 15 deterministic rubric checks and Elis passed 14 of 15. The only miss was a literal keyword check where the answer supplied success criteria without using the word "metric." Elis added measurable planning latency before the first task: 21.9 seconds on average in the ChatGPT-family run and 18.4 seconds in the Claude-family run. Task-cursor and settlement round trips were approximately 0.02 and 0.01 seconds in the measured probe.

Research questions

1. Does model right-sizing reduce dollar cost even when token volume does not fall?
2. Does the lower-cost execution arm preserve prompt-derived accuracy constraints?
3. What latency does private planning add before host execution begins?
4. Do agent creation, recall, automation tool naming, usage reporting, and saved-token settlement work through the MCP contract?

Method

Four prompts represented simple, moderate, complex, and deep work. Each host ran the same prompt set in a plain frontier-model arm and an Elis-directed arm. Elis called the local Token Saver MCP endpoint, followed every returned task cursor, and used the server's dynamically resolved model suggestion. Visible prompts and outputs were counted with `o200k_base`; this approximates Claude tokenization.

Provider prices came from the dated Elis model catalog and official catalog records used by the benchmark.

The ChatGPT-family run used manual constraint and coverage review. The Claude-family replication added deterministic prompt-derived checks for word limits, required sections, matrix structure, scenario coverage, rollback criteria, and pilot gates. It did not use a self-grading model as the primary accuracy metric.

Excluded usage includes host system instructions, MCP schemas and transport, hidden reasoning, planning-model and embedding usage, and tool calls. Therefore, reported token and cost values are visible-execution estimates, not provider invoices or complete end-to-end cost.

Results

Host family	Cases	Plain tokens	Elis tokens	Token change	Plain cost	Elis cost	Cost change
ChatGPT	4	6,451	9,075	+40.7%	$0.1845	$0.1176	-36.3%
Claude	4	2,966	2,109	-28.9%	$0.1413	$0.0235	-83.4%
Combined	**8**	**9,417**	**11,184**	**+18.8%**	**$0.3258**	**$0.1410**	**-56.7%**

The combined result demonstrates the central hypothesis: Elis can lower cost even when it uses more tokens. The mechanism is model right-sizing. The result also rejects a stronger, unsupported claim that orchestration always saves tokens. Task decomposition can duplicate context and output, while a lean graph can reduce both.

Accuracy

Complexity	Plain checks	Elis checks	Interpretation
Simple	4/4	4/4	Source-reported pass; plain word-count field records 122 and needs audit
Moderate	4/4	4/4	Eight criteria, matrix, and conditional recommendation satisfied
Complex	3/3	2/3	Elis missed a literal keyword but supplied substantive success criteria
Deep	4/4	4/4	TCO scenarios, gates, assumptions, and required coverage satisfied
Claude total	**15/15**	**14/15**	**Source-reported rubric totals; no observed substantive constraint regression**

The ChatGPT-family outputs were reviewed manually and are not combined into the numeric score. They were comparable on simple and moderate tasks, more detailed on the complex task, and partial on the deep task because independently generated subtasks duplicated content and disagreed on assumptions.

The Claude source record contains one scoring inconsistency: its plain simple case records `word_limit_120_body` as 122 while counting the case as passed. The table preserves the canonical source totals for reproducibility, but the exact 15/15 baseline score requires an audit against the retained output.

Latency

Measurement	ChatGPT host	Claude host
Mean plan creation	21.9 s	18.4 s
Plan creation range	Not retained per case	16.3-19.8 s
Next-task cursor probe	Not isolated	~0.02 s
Finish and settlement probe	Not isolated	~0.01 s

The plain arm has no Elis planning overhead. The measured 18-22 second delay is therefore the clearest current latency cost. Full host generation latency was not captured consistently and is not estimated. This evidence supports bypassing or short-circuiting the planner for trivial prompts while amortizing planning over complex, multi-step work.

Operational validation

Live MCP probes on `dev-18` verified that a first browser-automation prompt created a tenant candidate agent and an identical repeat recalled it with 1.0 match confidence. A browser screenshot task exposed the generic `browser` tool without leaking Elis's internal graph. The initial OpenClaw desktop probe failed because comma splitting produced degenerate tasks and omitted `computer` tools. Post-study regression verification now keeps the desktop goal as one coherent task and exposes the generic `computer` tool.

An earlier synthetic host-reported usage fixture persisted three task results, reported 3,150 tokens and $0.001335 actual estimated cost, compared them with a same-token baseline of $0.0345, and settled 3,876 saved tokens. However, the initial zero-usage automation probes exposed a settlement defect: cursor-advanced plans settled estimated savings despite having no execution usage. Post-study verification now settles those plans at zero with `estimated_only`, and task completion enters an expirable `awaiting_finish` state so an omitted finish call cannot reserve allowance forever. Fully catalog-priced execution now settles on measured same-token cost reduction, capped at the plan reservation; unpriced execution uses a capped token-volume fallback.

Price-selector audit and issue record

On 2026-08-12, live MCP plan `9d00ce8899e14ac7bd83951543cadf48` reproduced a high-severity selector defect: shared catalog helpers used the coarse `cost_weight` heuristic where their contract said "cheapest." In the OpenAI frontier tier this could recommend `o1` ($0.075 per equal 1K input and 1K output tokens) instead of `o3` ($0.010), an **86.7% controlled-comparison reduction** after correction. Commit `8290b06c` centralizes resolved hosted-price ordering; 51 focused tests passed. Full details and the remaining tool-cost issue are recorded in `docs/research/token-saver-issue-log-2026-08-12.json`.

Limitations

- One run per case cannot bound model or planner variance.
- The two host studies used different quality methods, so only the Claude rubric is numerically scored.
- The Claude source has a 122-word/pass inconsistency, so accuracy totals are source-reported rather than independently verified.
- Visible-token estimates exclude planning-model work, hidden reasoning, MCP framing, and tool usage; MCP telemetry now reports embedding representation and cost separately.

- Hosted search, file-search, container, browser, and computer-use charges are not included in model-token savings and need separate attested subtotals.
- Full end-to-end response latency and time to first token were not recorded consistently.
- The ChatGPT browser available during publication was logged out, so no new signed-in ChatGPT app run is claimed.
- Browser and OpenClaw validation covered planning contracts, not completed UI automation.
- Host-reported usage is not a cryptographic provider receipt. Catalog repricing and reservation caps reduce billing error, but paid settlement still needs an attested usage channel before adversarial clients are in scope.

Conclusions

Model right-sizing reduced estimated execution cost on both host families, but token savings depended on task-graph quality. Accuracy held at the deterministic rubric's resolution, with one keyword-level miss. Latency is the principal measured tradeoff: approximately 18-22 seconds of planning before the first task. The OpenClaw decomposition, zero-usage settlement, stranded-reservation, and same-token cost-accounting defects found during review now have focused regressions. The next study should use repeated independent executions, provider usage receipts, end-to-end timing, factual ground truth, and blinded model-judge panels as secondary evidence.

Reproduction and data

- Public summary dataset: `elis-token-saver-evidence-study.json`
- Source ChatGPT benchmark: `.codex/research/token-saver/benchmark-2026-08-13.json`
- Source Claude benchmark: `docs/research/token-saver-claude-benchmark-2026-08-13.json`
- MCP mounts tested: `/api/mcp/openai` and `/api/mcp/claude`