Asking a Second AI Only Helps When It Is Smarter: Seventy-Seven Controlled Runs on Agent Consultation, Blind Spots, and the Price of a Second Opinion

Zach Kellman
Actual Intelligence Labs, Sarasota, FL

August 30, 2026

Abstract

We ran 81 controlled agent runs on whether an AI should stop at a hard decision and ask a second AI what to do. Asking an equal never helped once. Putting a stronger model in charge of a weaker one scored 96.8 against 98.8 for the strong model working alone, using 2.5 percent of its tokens. Handing that director a structural map of the codebase pushed the best run to 98.82, an exact tie with the frontier model doing the entire job itself.

1 How Much of a Frontier Model Can You Get Without Paying for One?

> About 98 percent of its score for 2.5 percent of its tokens. A weaker model doing the work, with a stronger model making every hard call from a one-page brief, scored 96.8 where the strong model working alone scored 98.8. The director spent 5,390 tokens. Doing the job itself costs 217,000.

This is the finding. Everything else in this report is how we got here and what it cost us to believe it.

The task is a financial clearing engine defined by 38 interacting rules, plus a scheduling optimization with no clean answer. It was designed by a stronger model than any of the workers, specifically to resist being maxed out, and it is graded against a hidden answer key. A competent but lazy attempt scores 37.3.

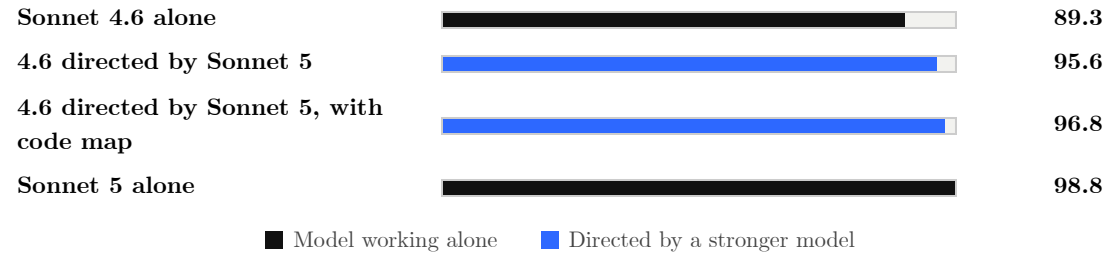

Figure 1: Clearing engine score out of 100, graded against a hidden answer key. Actual Intelligence Labs, 81-run consultation experiment, August 2026

Configuration	Mean	Best run	Frontier tokens per job
Sonnet 4.6 alone	89.33	93.77	0
4.6 directed by Sonnet 5	95.60	96.55	4,300
4.6 directed by Sonnet 5, with code map	96.81	98.82	5,390
Sonnet 5 alone	98.78	98.82	217,062

Table 1: The four configurations from Figure 1, with per-run detail. One task, one hidden-key grader, 4 runs each

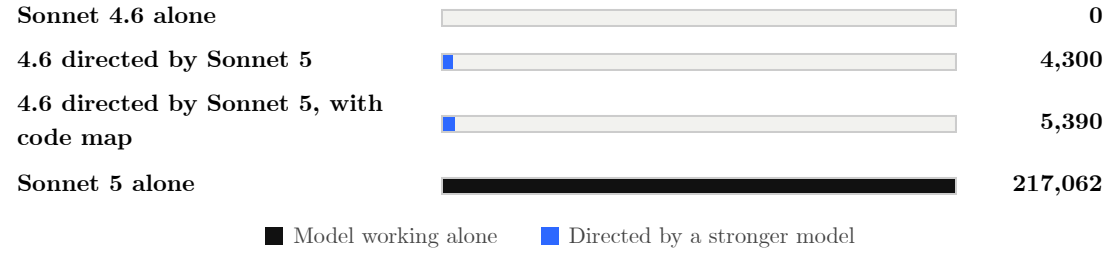

Figure 2: The same four configurations by frontier-model tokens consumed per job, linear scale. The two directed rows reach 96.8 and 98.8 percent of Sonnet 5's score on 2.0 and 2.5 percent of its frontier tokens. Actual Intelligence Labs, August 2026.

Read the third and fourth rows against each other. The directed configuration's best run scored 98.82. So did Sonnet 5's best run. Identical. One spent 5,390 frontier tokens getting there and the other spent 217,062.

2 Does Asking a Second AI Improve an Agent's Work?

> Not when that second AI is your own equal. Across four tasks with hidden graders, every configuration hit the same ceiling. On the one judgment task, a single fresh advisor ranked fourth of five and a three-advisor panel ranked last. The advice was fine. It changed nothing, and sometimes it cost us.

The protocol we tested is the strongest version of the idea. At every genuine crossroads, the working agent writes a full briefing: what the job is, what it has done, what it found, the fork it faces. It asks one open question. Never a yes or no, never an A-B-C-D. That brief goes to a model with

no context, no tools, no file access, and no memory of any prior question. The worker follows the answer. Then that advisor is destroyed and the next fork gets a brand new one.

We ran it against a control that just did the job, plus six other configurations: one advisor that remembers everything, three advisors at once, an advisor primed with an expert persona, an advisor handed a map of the codebase, an advisor whose ruling was binding, and a version where the agent writes the brief and then answers its own question.

Four of the five tasks ended in a dead heat. Every arm, every run, maximum score. Those are not weak instruments. On the invoice task, patching the existing code scored 80.6 and still passed every visible test. On the service task, fixing only the obvious file scored 61.1. Nobody took the bait. Not once.

The transcripts explain why. The agents did not out-think the traps. They out-verified them. They hand-computed expected answers straight from the spec, independent of their own code, then fuzzed the result against conservation invariants. When a job is specified tightly enough to check by running it, a careful agent turns the job into a verification problem, and verification saturates.

3 Why Do Fresh Eyes Miss the Blind Spot?

> Because the advisor only ever sees the letter. We buried five decision-critical facts in a business memo task, including a contract deadline that had already lapsed. Workers found all five before writing their briefs. The advisors found zero. A blind spot is by definition the thing that never makes it into the brief.

This is the part that kills the intuition. Fresh eyes are supposed to catch what tired eyes miss. But these fresh eyes never read the source material. They read a one-page summary written by the tired eyes.

So the advisor can only reason about what the worker already noticed. If the worker missed it, the advisor cannot catch it. If the worker caught it, the advisor is not adding anything. That is the whole trap, and it is structural.

There is a second cost. The worker has read every document. The advisor has one page. Its advice is sound and general, and a worker told to follow it trades what it actually knows for a stranger's defaults. That is how the consulted memos got worse. Judges ranked them blind, without knowing which configuration wrote which.

Configuration	**Mean rank (1 best, 10 worst)**
One advisor who remembers everything	3.17
No advisor at all	3.33
Write the brief, answer it yourself	4.33
Fresh advisor every time	7.50
Panel of three fresh advisors	9.17

Table 2: Business decision memo, ranked blind by three judges

The panel losing is its own lesson. Three independent advisors handed the same brief came back with nearly identical advice. You do not buy a second perspective. You buy an echo, three times.

4 When Does a Second Model Actually Help?

> When it outranks the worker. We put Sonnet 4.6 on the hands and a fresh context-free Sonnet 5 on the decisions, with its ruling binding. The weaker agent went from 89.3 to 95.6, and every directed run beat every solo run. It is the only configuration in the program that beat its own control.

One variable changed. Not the protocol, not the task, not the grading. Just who is on the other end of the phone. The worst directed run scored 94.05 and the best solo run scored 93.77. Nothing overlaps. Under an exact rank test the odds of that ordering by chance are about 1 in 35.

The mechanism shows up in the component scores. On raw rule-following both arms tied at 59 to 60 out of 60. The weaker model follows rules fine. The entire gap sits in strategy. Solo runs hand-built schedules costing 5,678 to 7,727. Directed runs were told to stop hand-building, turn their own verified engine into a simulator, and hill-climb from structurally different seeds. They landed between 5,016 and 5,629.

The stronger model never handed over an answer. It handed over a method. Method is exactly what fits in a one-page letter, which is why this channel works when nothing else did.

We also ran it backwards as a control. Strong model on the hands, weaker model giving binding orders. It barely moved, 98.78 down to 98.58. A weaker boss costs a strong worker almost nothing, because craftsmanship between the forks absorbs the bad direction. Capability only flows one way through this pipe.

5 What Does Giving the Director a Map of the Code Add?

> It fixes the blind-spot problem. We used Graphify to extract a structural map of the worker's own codebase and handed it to the director alongside the brief. Mean rose from 95.6 to 96.8, the best run tied the frontier model outright at 98.82, and rule-following hit a perfect 60 of 60 in all four runs.

The map is built from the source with a parser, not a model. It lists every symbol and everything that reaches it, with aliases and re-exports resolved, so a function imported under a different name still shows up as a caller of the original. It takes about a second to generate and adds roughly 1,800 tokens to a consultation.

Recall the structural problem from earlier: the director cannot catch what the worker did not write down. The map is the fix. It gives the director a channel into the work that does not pass through the worker's own account of it. We asked every run what the director flagged that they had not mentioned in their brief. All four caught something real:

- **Seven near-duplicate optimizer scripts**, none of them a source of truth, plus a separate file computing fees with its own independent function. The director warned the submitted cost figure might have come from the wrong code.

- **Two different scoring implementations** with different rejected-count logic, which the worker then cross-validated rather than trusting.
- **A search function that only shifted days by one or two**, meaning the worker's observed diminishing returns did not actually mean the space had been explored.
- **The full dependency chain** from the plan file down to the engine function, which is what drove the decision to verify the oracle before declaring the job done.

None of that was in any brief. A blind director could not have produced any of it. And the effect on correctness is the cleanest signal in the study: every graph-directed run scored a perfect 60 of 60 on the engine, while both the blind-director arm and the frontier model working alone each had a run slip to 58.97.

Honesty about the spread. The graph arm ranges from 92.78 to 98.82, wider than the blind arm, because in one run the director ordered the worker to stop optimizing and go harden the engine instead, which cost plan points. At four runs per arm the 1.2-point mean gain is directional, not significant. The perfect engine scores and the four-for-four blind-spot catches are the stronger evidence.

6 What Does This Actually Cost in Dollars?

> Frontier tokens drop 97.5 percent. Dollars drop 39 percent, and the gap between those two numbers is the worker's bill. The director costs about 1.7 cents, roughly 4 percent of the job. The other 96 percent is the cheap model doing the actual work.

Token counts are not money, so here is the money. At list prices Sonnet 5 is two dollars per million input and ten per million output. Sonnet 4.6 is three and fifteen. Haiku 4.5 is one and five. Agentic runs are input-heavy because the whole context gets resent every turn, so the figures below assume an 85 percent input mix and use our measured per-run token counts.

Configuration	Score	Worker tokens	Frontier tokens	Cost	Saving
Sonnet 4.6 alone	89.33	89,706	0	$0.431	38%
4.6 directed by Sonnet 5	95.60	89,706	3,599	$0.442	36%
4.6 directed by Sonnet 5, with code map	96.81	84,597	5,390	$0.423	39%
Sonnet 5 alone	98.78	0	217,062	$0.695	baseline
Haiku-class worker, directed (projected)	n/a	84,597	5,390	$0.153	78%

Table 3: The same four configurations again, priced at list rates. Saving is measured against Sonnet 5 alone

The shape of that bill is the point. In the directed configuration the worker spends 40.6 cents and the director spends 1.7. **Ninety-six percent of the cost is the cheap model doing the work,**

and four percent is the expensive model deciding what the work should be. You are not paying for intelligence by the hour. You are paying for it by the decision, and decisions are rare.

Now the part that should make you trust the number rather than doubt it. Sonnet 4.6 is not a cheap model. At list prices it costs more per token than Sonnet 5, three and fifteen against two and ten. Our worker is 1.5 times the price of the model bossing it around. The pricing is actively working against this result.

It still wins, because the directed worker burns far fewer tokens: about 85,000 against the 217,000 Sonnet 5 spends doing the job itself. Run the counterfactual where the worker is priced like Sonnet 5 instead of 1.5 times it, and the saving goes from 39 percent to 59 percent. Pair it with a genuinely cheap worker at Haiku rates and our measured token counts project 78 percent. We did not run that configuration; the last row of the table is arithmetic, not a result.

One more measured oddity worth stating. The directed setup came in at 42.3 cents against 43.1 for the same weak model working alone. **Directing it made it cheaper and 7.5 points better at the same time**, because a worker that is told what to do stops burning tokens flailing. That is a small difference on a small sample, so treat it as suggestive rather than banked.

The number that transfers to your stack is the frontier one: **2.5 percent of the expensive model's tokens bought 98 percent of its score.** Direction is cheap. Doing is expensive. That ratio is a property of the protocol, not of this model pair, and the dollar saving that comes with it gets better as your worker gets cheaper.

7 Can an AI Write a Question That Does Not Steer the Answer?

> Mostly not. Two independent auditors scored all 82 briefs for leading language. Roughly a third leaned toward an answer despite an explicit, itemized ban and workers actively trying to comply. The bias did not sit in the question. It sat in the background section above it.

The writers obeyed the rule exactly where you would audit for it. The options were parallel, equal length, no persuasive language, no ranking. Then they loaded the situation section with the facts supporting one option and left out the facts supporting the other. By the time you reach the choice, one branch is already dead.

Nothing in the question is unfair. The premises are. Lawyers do this deliberately. Agents do it by accident, and so do people. Tilt was statistically identical across every configuration, so it does not undermine the comparisons, but if you ship this pattern the brief needs auditing by something other than the thing that wrote it.

8 What Should You Actually Build?

> Put your cheap model on the hands and your best model on the decisions, three calls a job, ruling binding. Feed the director a structural map of the code so it can see past the

> worker's own account. Then spend what you saved on specification and verification, which is where every point in this study came from.

1. **Stop bolting advisors onto your best model.** Seven configurations tried it. None improved anything, and on judgment work it made the output worse.
2. **Invert it.** Cheap hands, expensive brain, binding rulings, three calls a job. That is the only thing in 81 runs that beat its own control, and it captured 98 percent of the frontier model's score.
3. **Give the director eyes.** A parser-built map of the codebase costs a second to generate and about 1,800 tokens to send. It caught a real structural defect the worker had not noticed in four out of four runs.
4. **Keep the letter, drop the stranger, when there is no gradient.** Writing the brief and answering it yourself cost nothing, never hurt, and beat real outside advice on the judgment task.
5. **Spend the savings on specification and verification.** Across five tasks the single best predictor of a perfect score was whether the job was defined tightly enough for the agent to check itself.

There is a bigger point buried in the four ceilings. We kept trying to prove that agents need help, and kept proving instead that a well-specified job does not need much. The market is spending heavily on orchestration layers to supervise models. Most of our results say the supervision was not the constraint. The specification was. The one exception is the one that pays: supervision is worth buying when it comes from something smarter than the thing being supervised.

A second opinion is a pipe. Water only moves through it if one end sits higher than the other.

9 Common Questions

Can a cheaper AI model match an expensive one? Close to it, if the expensive one makes the decisions. In our test a weaker model directed by a stronger one scored 96.8 where the stronger model working alone scored 98.8, and its best run tied at 98.82. The director used 2.5 percent of the tokens the strong model would have spent doing the job itself.

Should an AI agent ask another AI for a second opinion? Only if the second model is meaningfully stronger. Across 81 runs, consulting an equally capable model never improved an outcome and degraded judgment work. The gain only appears when there is a capability gap for the advice to flow down.

Why does a fresh AI advisor miss problems the worker missed? Because it only sees the brief the worker wrote. We planted five critical facts in a task; workers found all five and advisors found none. An advisor cannot catch what never made it into the summary, which is exactly what a blind spot is.

Does giving the advisor a codebase map help? Yes, and it is the fix for the blind-spot problem. A parser-built structural map gives the director a view that does not pass through the worker's own account. It raised the mean from 95.6 to 96.8, produced perfect rule-following in all four runs, and caught a real structural defect the worker had not mentioned every single time.

Is a panel of AI advisors better than one? No. Three independent advisors given the same brief returned nearly identical advice, and the panel produced the lowest-ranked deliverables in the study at three times the cost. You are buying redundancy, not perspective.

How much does model cascading actually save? Frontier-model token spend drops about 97.5 percent, but dollars drop about 39 percent, because the worker model still has to be paid. In our test the director was only 4 percent of the job's bill. With a genuinely cheap worker the projection is closer to 78 percent.

10 References

1. Anthropic. *Building Effective AI Agents*. https://www.anthropic.com/engineering/building-effective-agents
2. Graphify Labs. *Graphify: map a codebase into a queryable knowledge graph*. https://github.com/Graphify-Labs/graphify
3. Chen, Zaharia, Zou, arXiv. *FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance*. https://arxiv.org/abs/2305.05176
4. Shinn et al., arXiv. *Reflexion: Language Agents with Verbal Reinforcement Learning*. https://arxiv.org/abs/2303.11366
5. Madaan et al., arXiv. *Self-Refine: Iterative Refinement with Self-Feedback*. https://arxiv.org/abs/2303.17651
6. Zheng et al., arXiv. *Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena*. https://arxiv.org/abs/2306.05685

Primary research. Artifacts, hidden graders, per-run scores and all consultation transcripts were retained. Task authorship: Claude Fable 5. Workers: Claude Sonnet 5 and Claude Sonnet 4.6. Prepared by Zach Kellman, Actual Intelligence Labs, Sarasota, FL.