

---

# MarketBench: Evaluating AI Agents as Market Participants

---

**Andrey Fradkin\***  
Boston University & MIT IDE  
afradkin@gmail.com

**Rohit Krishnan**  
Independent Researcher

## Abstract

Markets are a promising way to coordinate AI agent activity for similar reasons to those used to justify markets more broadly. In order to effectively participate in markets, agents need to have informative signals of their own ability to successfully complete a task and the cost of doing so. We propose MarketBench, a benchmark for assessing whether AI agents have these capabilities. We use a 93-task subset of SWE-bench Lite, a software engineering benchmark, with six recently released LLMs as a demonstration. These LLMs are miscalibrated on both success probability and token usage, and auctions built from these self-reports diverge from a full-information allocation. A follow-up intervention where we add information about capabilities from prior experiments to the context improves calibration, but only modestly narrows the gap to a full-information benchmark. We also document the performance of a market-based scaffolding with these LLMs. Our results point to self-assessment as a key bottleneck for market-style coordination of AI agents.

## 1 Introduction

Agent benchmarks mostly ask whether a model can finish a task. Under market-style rules, agents typically name a price and convey how likely they are to succeed *before* a task is assigned, so allocation turns on self-reported odds and expected cost, not only on whether a run ultimately succeeds. We propose MarketBench, a benchmark for this self-assessment on realistic software-engineering tasks. Our initial instantiation uses 93 SWE-bench Lite tasks and six recently released LLMs. Models have weak self-assessment: current models are poor at forecasting both their success probability and their token usage, and those errors carry over into auction outcomes built from the same self-reports.

Why focus on this capability? Multi-agent systems are quickly becoming the standard way to do complex work through tools such as Claude Code and OpenClaw. However, these systems are complex and have ad-hoc rules for determining when and which types of agents are used across situations. In contrast, markets simplify complexity through the price mechanism. They are attractive for coordinating heterogeneous agents when task-specific fit, cost, or speed are local to the worker rather than visible to a central planner. But this logic only works if workers can say something informative about their own likely performance. Without calibrated self-assessment, bids are noisy from the start and downstream allocation suffers.<sup>2</sup> Even when the base model is public, the mapping from a task to success probability and token use is not predictable *ex ante*, and operator-side context (scaffolding, tools, state) makes private information genuinely decentralized.

---

\*Andrey Fradkin: Boston University and the MIT Initiative on the Digital Economy. Rohit Krishnan: Independent Researcher.

<sup>2</sup>This difficulty is compounded by what Dell’Acqua et al. (2023) call the “jagged technological frontier”: AI capability boundaries are irregular and unpredictable, with models excelling at some tasks while failing at others of seemingly similar difficulty.

MarketBench is based on SWE-bench Lite software-engineering tasks, and we intend to extend it to other tasks. In its current form, MarketBench consists of two task families:

1. **MarketBench Calibration:** We ask agents to forecast whether they can complete a task in one attempt and the total token usage they expect that attempt to consume. We then compare those reports to realized model runs and map the token forecasts into expected dollar costs using model-specific pricing.
2. **MarketBench Auction:** We use the previously elicited success and token estimates to derive bids for a reserve-price procurement simulation and measure the resulting payoffs relative to full information about capabilities and costs.

Separately, we present an illustrative scaffold experiment that elicits beliefs to route work across agents. We find that the scaffold benefits from using a diversity of agents, but is limited by the weak self-assessment of these agents. We treat this scaffold as a prototype of a system that can be improved in the future when agents improve their self-assessment capabilities.

The rest of this paper proceeds as follows. Section 2 reviews related work on multi-agent coordination, LLM self-evaluation, and market mechanisms for computational resource allocation. Section 3 presents a simple model of market-based task allocation among heterogeneous AI agents and derives the conditions under which market coordination outperforms fixed non-market routing rules. Section 4 introduces MarketBench, describing the task sets, bidding protocols, evaluation metrics, and benchmark results. Section 5 then presents our illustrative live scaffold experiment together with its empirical results. Section 6 discusses the implications for AI agent design, hybrid market–AI coordination, and the path toward market-ready agents.

## 2 Related Work

A growing literature proposes multi-agent LLM systems that decompose work across role-specialized agents and coordinate through explicit prompting, message passing, and hand-designed workflows (Wu et al., 2023; Hong et al., 2023; Li et al., 2023). Commercial coding agents similarly rely on centralized scaffolds that route subtasks, manage tool use, and implement retry logic (Anthropic, 2025; OpenAI, 2025). More dynamic orchestrators replace static chains with planner–scheduler architectures, but the core paradigm remains hierarchical: a central controller must decide decomposition, assignment, and failure recovery (Song et al., 2025; Tomašev, Franklin, and Osindero, 2026). Related work also explores market-inspired routing rules. For example, Alazraki et al. (2026) uses “strategy auctions” to score agent plans and route work among heterogeneous models, but this is not a literal market mechanism in our sense because bids do not arise from agents pricing their own expected success and cost under an incentive-compatible allocation rule. Our focus differs in asking whether coordination can be decentralized through market mechanisms, taking inspiration from how markets have been used at scale in society, and what agent capabilities such mechanisms presuppose.

The idea of bid-based allocation in multi-agent systems predates LLMs. Classical distributed AI introduced contracting and bidding protocols for task allocation, and the computer-systems literature developed market-based approaches to allocate fungible computational resources. Market-oriented programming, for example, computes allocations via competitive-equilibrium prices in artificial economies (Wellman, 1993). Those mechanisms work when agents can price their own costs and constraints accurately. In our setting, the traded object is heterogeneous cognitive labor whose quality and cost depend on model, context, and tools, so meaningful bids require metacognitive self-assessment.

A second, related, literature evaluates frontier agents on complex tasks. METR characterizes the time horizon over which an agent can sustain autonomous work (Kwa et al., 2025); GDPval measures performance on economically meaningful occupational tasks (Patwardhan et al., 2025); and the Holistic Agent Leaderboard standardizes comparisons across models and scaffolds and uses rollout logs to surface behavioral failure modes (Kapoor et al., 2025). Our closest reference is Rabanser et al. (2026), which argues that single success metrics obscure important dimensions of agent performance and elicits the model’s confidence that its answer is correct, evaluating those forecasts using the Brier score, as we do. Our elicitation target differs: we ask for ex ante beliefs about task-level success and expected cost, rather than confidence in the correctness of a realized answer.

This connects naturally to work on LLM confidence elicitation and calibration. Numerous papers document miscalibration and elicitation sensitivity (Xiong et al., 2024; Geng et al., 2024; Zhang et al., 2024). We extend this agenda from answer-level confidence to agentic production: market participation requires calibrated beliefs about task completion and expected cost, beyond correctness alone. Finally, our motivation is grounded in classic arguments for decentralized coordination. Hayek emphasizes that prices aggregate dispersed local knowledge (Hayek, 1945).

### 3 Conceptual Framework

In this section, we walk through a stylized model in which the value of a market mechanism for a principal looking to complete a task comes from task-specific private information. We consider this framework an intuition pump, rather than a full model of a market-based scaffolding.

A principal receives value  $v > 0$  if a task is completed. There are two agents,  $H$  and  $L$ . Agent  $H$  is the “better” agent on average but is also more expensive: it has higher baseline capability  $a_H > a_L$  and higher cost  $c_H > c_L > 0$ . Each task has difficulty  $d$ . For agent  $i \in \{H, L\}$ , realized task fit is  $a_i + \varepsilon_i$ , where  $\varepsilon_i$  is an idiosyncratic draw. The agent succeeds if  $a_i + \varepsilon_i \geq d$ . If  $F$  denotes the cumulative distribution function of  $\varepsilon_i$ , then the ex ante completion probability of agent  $i$  on a task of difficulty  $d$  is

$$p_i(d) = 1 - F(d - a_i),$$

so  $p_H(d) \geq p_L(d)$  for every  $d$ .

Three non-market baselines ignore the task-specific draw  $\varepsilon_i$ : assign only the better agent, only the worse agent, or both in parallel. Their expected welfare is  $W_H(d) = vp_H(d) - c_H$ ,  $W_L(d) = vp_L(d) - c_L$ , and  $W_P(d) = vQ_P(d) - c_H - c_L$  with  $Q_P(d) = 1 - (1 - p_H(d))(1 - p_L(d))$ . The better agent dominates the worse one when  $v[p_H(d) - p_L(d)] > c_H - c_L$ , and parallel execution beats either single-agent rule only when its incremental success probability covers the redundant cost.

Now suppose that before bidding each agent observes its own task-specific draw  $\varepsilon_i$  and hence learns whether the task lies within its current capability frontier. To keep the algebra transparent, assume that after observing  $\varepsilon_i$  the agent knows whether it can complete the task, so its realized completion probability is

$$\pi_i(d, \varepsilon_i) = 1\{a_i + \varepsilon_i \geq d\}.$$

Consider a success-contingent procurement contract that pays  $b_i$  only if the task is successfully completed. Agent  $i$ 's expected payoff from such a contract is  $\pi_i(d, \varepsilon_i)b_i - c_i$ , so a zero-profit bid is  $b_i = c_i/\pi_i(d, \varepsilon_i)$ . The principal therefore awards the task to the cheapest agent among those who know they can solve it, provided the implied payment is below  $v$ .

Under independence, this yields a simple expression for expected value. Let  $p_i(d)$  denote the ex ante probability that agent  $i$  declares itself able to solve the task. Since the worse agent is cheaper, it wins whenever it is capable; the better agent is used only when the worse agent cannot solve the task and the better one can. Hence the market allocation has expected value

$$W_M(d) = p_L(d)(v - c_L) + (1 - p_L(d))p_H(d)(v - c_H),$$

expected cost

$$C_M(d) = p_L(d)c_L + (1 - p_L(d))p_H(d)c_H,$$

and completion probability

$$Q_M(d) = p_L(d) + (1 - p_L(d))p_H(d).$$

The market allocation is valuable because it conditions on private, task-specific information rather than on average success rates alone.

To compare the welfare of the different allocation rules more transparently, it is useful to work directly with ex post capability states. Let  $\lambda_{\ell h}(d) = \Pr(\pi_L = \ell, \pi_H = h)$  for  $\ell, h \in \{0, 1\}$ , where the first index refers to  $L$  and the second to  $H$ . Thus  $\lambda_{10}$  is the probability that only  $L$  can solve the task,  $\lambda_{01}$  is the probability that only  $H$  can solve it,  $\lambda_{11}$  is the probability that both can solve it, and  $\lambda_{00}$  is the probability that neither can solve it. The expected welfare of the market mechanism can then be written as

$$W_M(d) = \lambda_{10}(v - c_L) + \lambda_{01}(v - c_H) + \lambda_{11}(v - c_L),$$

since the market picks the cheapest capable agent in each state and does not allocate the task when neither agent can solve it.

**Proposition 1.** *Suppose  $v > c_H > c_L > 0$ . Then the market allocation weakly dominates each non-market alternative. Moreover, the dominance is strict whenever there is positive probability of a state in which the alternative either allocates the task to the wrong agent, pays a redundant agent, or pays for an attempt when no agent can solve the task.*

Relative to fixed assignment, the market preserves the good states while correcting the bad ones: it uses  $L$  when only  $L$  can solve, uses  $H$  when only  $H$  can solve, uses the cheaper agent when both can solve, and avoids wasting cost when neither can solve. Relative to parallel execution, the market achieves the same completion outcome in every state but avoids paying redundant costs. The proof is in Appendix B.

The assumption that agents perfectly observe  $\varepsilon_i$  is stronger than what practice requires. In realistic systems, agents only need a signal about likely success or cost that is more informative, or at least not fully redundant, relative to the principal’s own signal. If an agent’s self-assessment is just a noisy restatement of what the principal already knows, bidding adds little. But if self-assessment contains even modest incremental information about task-specific fit, then bidding can improve allocation and reduce unnecessary parallelism.

## 4 MarketBench

MarketBench is a benchmark for AI agents’ readiness for market participation. Our initial instantiation uses SWE-bench Lite tasks (Jimenez et al., 2024): real GitHub issue-fix pairs with executable test suites and a binary notion of success, drawn from a small set of mature open-source Python repositories so that per-repository performance priors are well-defined. The novelty of MarketBench lies in what we ask agents to reveal about those tasks before allocation. We study whether models can forecast their own success probabilities and expected resource use well enough to support decentralized routing.

### 4.1 Calibration

The calibration task asks a model to inspect a software-engineering task and forecast whether it can solve that task in one attempt, together with the total token budget it expects to consume. Before the solve attempt, the model is prompted as follows:

Estimate the probability that you could complete this task correctly in one attempt. Return JSON only with fields `p_success` (0..1), `estimated_tokens_total` (total model tokens for one full solve attempt), `rationale` (optional).

We compare the reported `p_success` to realized pass/fail outcomes and map token forecasts into expected dollar costs using model-specific blended pricing. Phase I uses 93 SWE-bench Lite tasks and six frontier models, yielding 558 model-task rows. The elicitation prompt exposes the task ID, title, full problem statement, and acceptance commands, and responses are collected as strict JSON at temperature 0. Realized success labels come from strong-scaffold external SWE-bench runs. By “external scaffold,” we mean a stronger SWE-bench execution environment with interactive shell access, direct test execution and feedback, structured file editing, and multi-turn revision.

### 4.2 Auction

The second task family asks whether those self-assessments are good enough to support bidding. We study a simple reserve-price procurement simulation in which each model’s bid is mechanically derived from its previously elicited success probability and token-cost estimate rather than from a separate bidding prompt. A given auction only includes one model as a participant, but is incentive-compatible given that it is a second-price auction with an independently drawn reserve.

The model-specific breakeven bid is

$$b^* = \frac{\text{token cost} + \text{penalty} \times (1 - p_{\text{success}})}{p_{\text{success}}}$$

Table 1: Calibration on 93 SWE-bench Lite tasks. The pass-rate spread is much narrower than the confidence spread. Only Claude Opus 4.5 and Claude Sonnet 4.5 achieve positive Brier skill relative to a naive base-rate forecast. The final two columns summarize the magnitude of token-estimation error: estimated tokens are the model’s forecast, and realized tokens are the actual usage implied by dollar cost and model-specific pricing.

Model	Mean $p$	Pass rate	Skill	Est. toks	Realized toks
Claude Opus 4.5	76.5%	80.6%	+0.060	14,774	35,820
Gemini 3 Pro Preview	92.9%	80.6%	-0.111	1,801	55,969
GPT-5.2	63.2%	80.6%	-0.195	2,909	25,036
GPT-5.2-pro	66.3%	79.6%	-0.111	2,647	3,970
Claude Sonnet 4.5	75.8%	77.4%	+0.018	18,333	53,085
GPT-5-mini	61.4%	75.3%	-0.305	2,595	28,276

The simulation uses model-specific blended token pricing,<sup>3</sup> a failure penalty of \$2, and reserve prices drawn independently from a Uniform[0, 1] distribution, with 100 reserve draws per row and deterministic seeds. Expected profit uses the model’s reported `p_success`; realized profit evaluates the same mechanically derived bid against the observed pass indicator.

### 4.3 Calibration Results

We begin by evaluating whether models can predict their own task success on 93 SWE-bench Lite tasks. Table 1 reports mean stated success probability, realized pass rate, Brier skill relative to a base-rate benchmark, and two summary statistics for token forecasting.

The main fact from Phase I is miscalibration. All six models cluster between 75.3% and 80.6% realized pass rates, yet their average stated success probabilities range from 61.4% to 92.9%. Gemini 3 Pro Preview is sharply overconfident, while GPT-5.2 and GPT-5-mini are underconfident relative to their realized pass rates. To support the scaffold case study below, we select a common subset of 50 tasks that all six models attempted; on that slice, GPT-5.2 is the strongest single-model baseline on the external scaffold at 37/50 tasks, while a simple oracle that always picks the best model reaches 42/50. Token self-estimates are also severely understated: the median estimated-to-actual token ratio is 0.2.

We also assessed whether prompt modification can partially address this bottleneck. For a given task, we computed performance on the other SWE-bench tasks and prepended it to the prompt. The performance metrics we included were pass rate, average stated confidence, and typical token underestimation; Appendix A shows a representative example, and Table A1 reports the aggregate before-and-after calibration summary. On the full six-model rerun, the mean Brier score improves from 0.1835 to 0.1693 and ECE improves from 0.1065 to 0.0616; token forecasts also become less severely understated. Even a simple self-history card makes the forecasts more useful.

### 4.4 Reserve-Price Auction Results

We next ask whether these self-assessments, though noisy, are nevertheless sufficient for profitable bidding in the reserve-price auction described above. Table 2 summarizes simulated outcomes on the same 93-task calibration set.

Two patterns stand out. First, all models earn less than the oracle benchmark, often by a wide margin. GPT-5.2, for example, earns roughly \$0.006 per task in realized profit versus \$0.385 for its oracle counterpart. Second, poor calibration distorts allocation even when average profits remain positive. Gemini wins 84.6% of simulated auctions and earns the largest realized profit, but this dominance is driven by aggressive bidding despite the weak calibration signal documented above. The resulting auction is therefore far from the full-information outcome.

The self-knowledge follow-up from the preceding subsection partly explains why. It improves the underlying forecasts and shifts allocation in the predicted direction, but the aggregate auction num-

<sup>3</sup>Blended pricing is a single dollars-per-token rate for each model: we divide the run’s total dollar cost by its total token count (input and output pooled), so all token categories are valued at that common average.

Table 2: Single-model reserve-price auction outcomes. Profits are reported in dollars per task under the reserve-auction simulation. The oracle column assumes perfect knowledge of which tasks are actually solvable and abstention on the rest.

Model	$N$	Win rate	Exp. profit	Realized profit	Oracle profit
Claude Opus 4.5	93	28.2%	\$0.077	\$0.080	\$0.260
Claude Sonnet 4.5	93	29.3%	\$0.062	\$0.064	\$0.288
Gemini 3 Pro Preview	93	84.6%	\$0.353	\$0.303	\$0.391
GPT-5-mini	93	21.6%	\$0.054	\$0.050	\$0.375
GPT-5.2	93	5.4%	\$0.005	\$0.006	\$0.385
GPT-5.2-pro	93	5.3%	\$0.007	\$0.007	\$0.227

bers move only at the margin (mean realized profit \$0.085 to \$0.082, oracle gap 0.24 to 0.23). Prompt-level self-knowledge is enough to repair *who* wins; it is not yet enough to repair *how much* the principal recovers. This section does not report a full reserve-distribution or payment-rule sensitivity sweep. Later informed competitive auctions suggest that prompt framing changed bidding behavior more than payment-rule variation changed allocation accuracy. The deeper bottleneck still appears to be weak task-by-task self-assessment.

## 5 Illustrative Scaffold Experiment

### 5.1 Setup

Beyond the benchmark, we built a market-inspired scaffold to test whether model-submitted bids improve allocation in a multi-agent setting. We call it the *live scaffold*, contrasted with the *external scaffold* used for benchmark labels and single-model baselines. The live scaffold treats each SWE-bench issue as a single task and verifies a submitted patch at the end. Each attempt is one shot — a single patch, with no interactive shell, no test-feedback loop, and no within-attempt revision — but each task gets up to two attempts, and a hard worker-exclusion rule forces the second attempt to a different model. The scaffold is operator-run: the operator chooses assignments, pays model-run costs directly, and uses a score as an internal routing heuristic, so it is market-like rather than a literal procurement auction with outside suppliers. We treat it as a case study in routing under imperfect self-assessment rather than an implementation of the success-contingent procurement contract analyzed in Section 3.

The scaffolding works as follows. Each available worker submits an ask, a self-assessed probability of success, and an expected completion time, although the current implementation does not yet use the time field in the routing score (it is used only as a tie-breaker). We model the worker as expecting to receive its ask minus its cost upon winning and successfully executing the task, and paying a penalty if it fails.

The operator pays execution costs and decides whether to assign a task and to whom based on a score. For bidder  $i$ ,

$$\text{Score}_i = p_i \times (\text{Utility} - \text{Ask}_i) - (1 - p_i) \times \text{Penalty}_i - E[\text{Cost}_i].$$

where Utility is the operator’s per-task utility (a scaffold-level constant),  $\text{Ask}_i$  is bidder  $i$ ’s requested payment on success, and  $\text{Penalty}_i$  is the scaffold’s explicit loss when bidder  $i$ ’s attempt fails.<sup>4</sup>

All bids with positive scores remain eligible, and the highest-scoring available worker receives the task. If that worker fails, the engine considers the next-highest eligible bid. Tasks are run with the same six frontier models used in the benchmark, a utility of 90 internal credits for every task, an explicit failure-penalty settlement rule, a 90-second bidding window, a 900-second execution limit, and at most two attempts per task. Workers are required to submit bids on tasks, and a worker that fails a task is excluded from retrying that same task.

<sup>4</sup>In the implementation, the failure penalty is itself a function of the bidder’s stated  $p_i$ :  $\text{Penalty}_i = \rho \cdot \text{Utility} \cdot (0.5 + p_i)$ , with  $\rho$  a fixed scaffold parameter. Overconfident bidders therefore face a heavier expected loss on failure.

Table 3: Illustrative live scaffold experiment results on 50 common tasks. The same-scaffold comparison keeps the task slice, verifier, timeout, and two-attempt cap fixed, but not the worker pool: the market condition uses six workers and diverse-model retry, whereas the solo condition always uses GPT-5.2.

Execution paradigm	Pass rate	Passes
Market (our scaffold)	58.0%	29/50
Solo GPT-5.2 (our scaffold)	48.0%	24/50
Solo GPT-5.2 (external scaffold)	74.0%	37/50
Oracle ceiling (external scaffold)	84.0%	42/50

Table 4: Alternative Scaffold Specifications. All rows use the common 50-task slice. The matched centralized-router and matched market rerun keep the same six workers, verifier, and 900-second execution limit. The self-knowledge market row changes only the private bidding context by adding a held-out calibration prior and bidding guardrails before bidding. The Codex row changes both execution path and budget, so it is a diagnostic rather than an apples-to-apples replacement for the published benchmark.

Follow-up	Pass rate	Passes
Matched centralized router	54.0%	27/50
Matched market rerun	46.0%	23/50
Market with self-knowledge bidding prior	56.0%	28/50
Codex + GPT-5.2 (1800s budget)	70.0%	35/50

## 5.2 Scaffold Performance and Extensions

We evaluate the live scaffold on the common 50-task subset introduced above. Table 3 reports the published market-versus-solo comparison, which is best read as a scaffold-level comparison rather than as a clean mechanism test. “Market (our scaffold)” is the six-worker live-routing condition described above. “Solo GPT-5.2 (our scaffold)” uses the same internal scaffold but always assigns GPT-5.2, including on retry. “Solo GPT-5.2 (external scaffold)” is the stronger SWE-bench baseline from Section 4, and the oracle ceiling asks what pass rate an omniscient router could reach on the external scaffold by always picking the right model for each task.

The market-versus-solo comparison keeps the task slice, verifier, 900-second execution limit, and two-attempt cap fixed, but not the worker pool: the market condition accesses a six-model pool and can route a retry to a different model, whereas the solo condition always uses GPT-5.2. Under those conditions, the market outperforms the same-scaffold solo baseline by 5 tasks (10pp); 18 tasks are solved by both, 11 by the market only, 6 by solo GPT-5.2 only, and 15 by neither. That pattern is consistent with gains from model diversity and retry, but the p-value is 0.3, and as the matched-rerun row in Table 4 shows (29/50 vs. 23/50 for the same nominal condition), run-to-run variation at  $n = 50$  is on the same order as the across-scaffold differences we are trying to measure. We therefore treat the scaffold-level comparison as suggestive and rest the paper’s main claims on the calibration evidence in Section 4.

Table 3 also demonstrates the underperformance of the live scaffold relative to external scaffolds: the same GPT-5.2 loses 26 percentage points (13 tasks, 74% vs. 48%) when moved from the external scaffold to ours. The reason for this difference is that the external scaffold gives the model an interactive shell, direct test execution and feedback, multi-turn iteration, and structured file-editing tools; the live scaffold offers none of these and limits each worker to a single patch (raw diff or BEGIN\_FILE block) per attempt. The market recovers about 10 percentage points of that gap through model diversity; the rest would require scaffold upgrades, not better bidding.

The market run also used more compute than solo GPT-5.2, because multiple workers inspect each task before clearing and some tasks take a second attempt: about 5.82M total tokens for the market run versus 4.37M for the solo baseline (200.8K versus 182.2K tokens per successful pass).

There are two explanations for why our market scaffold can beat a solo model: the market-clearing rule, and the diverse worker pool.

To separate these explanations, we ran a follow-up that keeps everything identical, but replaces the market-clearing rule with a centralized router. The router is a single LLM call (GPT-5.2-pro) that reads the task descriptions, the available workers with their model identifiers, expected cost hints, retry-exclusion constraints, and other available context such as discussion notes and worker reputation summaries, and then directly assigns a worker to each task. The assigned worker still executes the task through the same path as in the market condition; the only difference is who chooses the assignment. Under those matched conditions, the centralized router reaches 27/50 while the market reaches 23/50. The gain from the published market comparison therefore appears to come primarily from model diversity rather than from the present bidding rule. This shouldn't be surprising, since models perform poorly in self-assessment as measured by MarketBench.<sup>5</sup>

If overconfident bidding is the bottleneck, can better priors help? To test this, we ran a second follow-up that keeps the market-clearing rule but changes only the private bidding context: each worker receives a held-out calibration prior before bidding, summarizing its own historical pass rate and overconfidence tendency. The hard-prior rule (for the related calibration card see Appendix A) starts from the held-out pass rate minus any historical overconfidence gap, then asks the worker to raise `p_success` only when the current task gives direct evidence that it is easier than the held-out average. Under that intervention, the market reaches 28/50—up from 23/50 in the matched rerun and slightly above the centralized router at 27/50. This is suggestive that better self-knowledge would yield even greater performance benefits.

**Sensitivity to execution harness and time budget.** A separate question is how much the live scaffold's weaker execution environment constrains all of these results. To probe this, we ran a diagnostic on the same 50-task slice using GPT-5.2 through a different execution route, called the `codex-direct` worker path, with the per-task limit doubled from 900 to 1800 seconds. Under those changed conditions, the follow-up reaches 35/50 passes, compared to 23–29/50 for our market scaffold and 24/50 for solo GPT-5.2 in our scaffold. The cost picture is very different, however. The `codex-direct` run consumes 321.3 million total tokens, versus 4.37 million for the solo GPT-5.2 run and 5.82 million for the market run — roughly 55–75× more compute for, at most, a 12-task improvement over our market scaffold and an 11-task improvement over solo GPT-5.2.

Two observations follow. First, when comparing models or scaffolds, the execution path is first order: routing and bidding choices that move pass rates by a few tasks are dwarfed by execution-environment effects. Second, the steep token cost of those gains is itself a reason to care about market-style coordination: if substantial pass-rate improvements available at the frontier come with order-of-magnitude compute increases, then mechanisms that route only the tasks that need that budget to the expensive path become economically important rather than cosmetic.

## 6 Discussion

As the number and variety of AI agents being used in production increases exponentially, identifying ways for them to efficiently and successfully collaborate on hard problems becomes a key bottleneck. For human agents, markets emerged as the dominant mechanism for this kind of coordination, and there are reasons to expect a similar role for AI agents.

Yet current agents do not know enough about their own production functions. They can solve complex tasks, but they cannot reliably say in advance which tasks they can solve, how likely they are to succeed, and what the attempt will cost. But that information is exactly what a market is supposed to aggregate. If bids do not reflect informed private information, the price system cannot do its job. Even though a brief summary of a model's own past behavior materially improves calibration, it is still far from the theoretical maximum.

These observations suggest that the optimal solution for coordinating agents will likely be a combination of markets and AI. Markets are useful when knowledge is local, dispersed, and hard to centralize. That description fits agentic work rather well. Different models have different strengths, different tool use, different failure modes, and different costs. But with imperfect self-knowledge and behavior, agents may not be reliable in their own assessments of their abilities. The role of AI

---

<sup>5</sup>The main driver of the market's drop in the matched rerun is Gemini. Gemini bids aggressively, so the market selects it more often. But its realized performance is weak. This is a bid-calibration problem: an overconfident worker crowds out better-suited alternatives.

in such a system is to augment the market, scoring agent bids based on more centralized sources of information, especially about the trustworthiness of an agent’s bids.

An analogy to other types of auctions is useful. In procurement, buyers often care about more than price. They care about quality, speed, reliability, and other attributes of performance. Scoring auctions handle that problem directly: the winner is the bidder offering the best combination of price and non-price characteristics (Che, 1993). Similarly, ad auctions combine this idea with a statistical model of relevance (Varian, 2007).

A mature market for AI agents is likely to work similarly. An agent that is cheap but unreliable should have a low quality score. In contrast, an agent that is more expensive but bids accurately may be the efficient choice. When multiple agents bid, a scoring function should appropriately weigh the bids to determine the winner. Concretely, the quality score in such a market could combine an agent’s held-out calibration record on a benchmark like MarketBench (how well its stated  $p_{\text{success}}$  has tracked realized outcomes), domain-specific reputation aggregated across past tasks of similar type, and, where verification is possible, the rate at which a third-party verifier accepted its outputs. These are exactly the inputs that a centralized router currently uses ad hoc; making them explicit and tradeable across operators is what would turn the existing scaffolds into a market.

Recent evaluations also show that measured agent performance keeps improving at much larger inference budgets than standard setups typically allow (AI Security Institute, 2026), and our own codex-direct diagnostic points the same way. If frontier pass-rate gains come from spending more compute per task, a single scalar bid quoting one expected cost is the wrong object: the natural next step — once scalar self-assessment is reliable enough to build on — is to elicit a mapping from token budget to expected success, so the principal can buy the right point on each agent’s cost-quality frontier.

The agenda that we propose has the following next steps. First, self-assessment should become a target of training and evaluation in its own right. Calibration, abstention, and budget discipline in bidding are core capabilities for decentralized coordination. Second, richer market institutions will need to arise, including reputation, escrow, and other mechanisms. Third, this benchmark should be extended beyond software engineering into other complex settings.

## References

- AI Security Institute. 2026. “Evidence for inference scaling in AI cyber tasks: Increased evaluation budgets reveal higher success rates.” AISI Blog. <https://www.aisi.gov.uk/blog/evidence-for-inference-scaling-in-ai-cyber-tasks-increased-evaluation-budgets-reveal-higher-s>
- Alazraki, Lisa, William F. Shen, Yoram Bachrach, and Akhil Mathur. 2026. “Scaling Small Agents Through Strategy Auctions.” *arXiv preprint arXiv:2602.02751* <https://arxiv.org/abs/2602.02751>.
- Anthropic. 2025. “Claude Code.” <https://docs.anthropic.com/en/docs/claude-code>.
- Che, Yeon-Koo. 1993. “Design Competition Through Multidimensional Auctions.” *The RAND Journal of Economics* 24 (4): 668–680.
- Dell’Acqua, Fabrizio, Edward McFowland III, Ethan R. Mollick, Hila Lifshitz-Assaf, Katherine Kellogg, Saran Rajendran, Lisa Kraye, François Cadelon, and Karim R. Lakhani. 2023. “Navigating the Jagged Technological Frontier: Field Experimental Evidence of the Effects of AI on Knowledge Worker Productivity and Quality.” *Harvard Business School Technology & Operations Mgt. Unit Working Paper* (24-013).
- Geng, Jiahui, Fengyu Cai, Yuxia Wang, Heinz Koepl, Preslav Nakov, and Iryna Gurevych. 2024. “A Survey of Confidence Estimation and Calibration in Large Language Models.” .
- Hayek, Friedrich A. 1945. “The Use of Knowledge in Society.” *The American Economic Review* 35 (4): 519–530.
- Hong, Sirui, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. 2023. “MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework.” *arXiv preprint arXiv:2308.00352* .

- Jimenez, Carlos E., John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. 2024. “SWE-bench: Can Language Models Resolve Real-world GitHub Issues?” In *The Twelfth International Conference on Learning Representations*. <https://openreview.net/forum?id=VTF8yNQM66>.
- Kapoor, Sayash, Benedikt Stroebel, Peter Kirgis, Nitya Nadgir, Zachary S. Siegel, Boyi Wei, Tianci Xue, Ziru Chen, Felix Chen, Saiteja Utpala, et al. 2025. “Holistic Agent Leaderboard: The Missing Infrastructure for AI Agent Evaluation.” *arXiv preprint arXiv:2510.11977* <https://arxiv.org/abs/2510.11977>.
- Kwa, Thomas, Ben West, Joel Becker, Amy Deng, Katharyn Garcia, Max Hasin, Sami Jawhar, Megan Kinniment, Nate Rush, Sydney Von Arx, et al. 2025. “Measuring ai ability to complete long tasks.” *arXiv preprint arXiv:2503.14499*.
- Li, Guohao, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. “CAMEL: Communicative Agents for “Mind” Exploration of Large Language Model Society.” *arXiv preprint arXiv:2303.17760*.
- OpenAI. 2025. “Codex.” <https://openai.com/index/introducing-codex/>.
- Patwardhan, Tejal, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simón Posada Fishman, Marwan Aljubei, Phoebe Thacker, Laurance Fauconnet, et al. 2025. “GDPval: Evaluating AI Model Performance on Real-World Economically Valuable Tasks.” *arXiv preprint arXiv:2510.04374* <https://arxiv.org/abs/2510.04374>.
- Rabanser, Stephan, Sayash Kapoor, Peter Kirgis, Kangheng Liu, Saiteja Utpala, and Arvind Narayanan. 2026. “Towards a Science of AI Agent Reliability.” *arXiv preprint arXiv:2602.16666* <https://arxiv.org/abs/2602.16666>.
- Song, Xinyuan, Zeyu Wang, Siyi Wu, Tianyu Shi, and Lynn Ai. 2025. “Gradientsys: A Multi-Agent LLM Scheduler with ReAct Orchestration.” *arXiv preprint arXiv:2507.06520* <https://arxiv.org/abs/2507.06520>.
- Tomašev, Nenad, Matija Franklin, and Simon Osindero. 2026. “Intelligent AI Delegation.” <https://arxiv.org/abs/2602.11865>.
- Varian, Hal R. 2007. “Position Auctions.” *International Journal of Industrial Organization* 25 (6): 1163–1178.
- Wellman, Michael P. 1993. “A market-oriented programming environment and its application to distributed multicommodity flow problems.” *Journal of artificial intelligence research* 1: 1–23.
- Wu, Qingyun, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xu Zhang, Siru Wang, Shujian Zhang, et al. 2023. “AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation.” *arXiv preprint arXiv:2308.08155*.
- Xiong, Miao, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. 2024. “Can LLMs Express Their Uncertainty? An Empirical Evaluation of Confidence Elicitation in Large Language Models.” *arXiv preprint arXiv:2306.13063*.
- Zhang, Mozhi, Mianqiu Huang, Rundong Shi, Linsen Guo, Chong Peng, Peng Yan, Yaqian Zhou, and Xipeng Qiu. 2024. “Calibrating the Confidence of Large Language Models by Eliciting Fidelity.” In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

## A Self-Knowledge Card for Direct Calibration

The self-knowledge calibration follow-up prepended a short held-out history card before the standard direct calibration prompt. The exact numbers varied by model and by the held-out tasks available for that model.

SWE-bench instances are drawn from real open-source repositories, not synthetic toy problems. Task identifiers follow the usual convention: for example, `sympy__sympy-17630` denotes an issue

Table A1: Aggregate six-model calibration summary with the self-knowledge card. Both rows aggregate the same six-model, 93-task setup rather than reporting a single model. “Skill” is aggregate Brier skill relative to a base-rate benchmark. “Est./actual toks” is the median estimated-to-actual token ratio, so larger values indicate less severe underestimation. The corresponding ECE values, reported in the text, improve from 0.1065 to 0.0616.

Prompt condition	Mean $p$	Pass rate	Skill	Est./actual toks
Direct calibration prompt	72.7%	79.0%	−0.107	0.1929
Self-knowledge card + direct prompt	80.6%	79.0%	−0.022	0.2501

from the `sympy/sympy` repository. The main experiments use a fixed 93-task subset of MarketBench; that subset contains multiple tasks from some repositories, rather than a single task per repository. When we form the self-knowledge card for a given evaluation task, the held-out statistics exclude the current task. The figure “across 92 held-out tasks” aggregates all other tasks in this setup for the same model, spanning every repository represented in the subset. The line “on prior tasks from `sympy/sympy` (21 held-out tasks)” is the same-repository slice of that pool: among those 92 tasks, the ones that also come from `sympy/sympy`. The repository-specific line was included only when there was enough held-out data for that repository.

The template for the card was:

Use the historical self-knowledge summary below as a prior for this model.  
 These statistics come from held-out MarketBench tasks for this same model and exclude the current task.  
 Historical self-knowledge summary:

- Across  $\{N\_held\}$  held-out tasks, your pass rate was  $\{pass\_rate\}$ .
- Your mean previously stated success probability was  $\{mean\_p\}$ ; you were historically  $\{over\}$  or  $\{under\}$ confident by  $\{gap\}$ .
- Your actual solve-token usage was typically  $\{ratio\}x$  your estimate.
- (*Included when  $\geq 5$  held-out tasks from the same repository were available.*) On prior tasks from  $\{repo\}$  ( $\{N\_repo\}$  held-out tasks), your pass rate was  $\{repo\_pass\_rate\}$  and your mean stated success probability was  $\{repo\_mean\_p\}$ .

Start from these historical tendencies, then update using the specific evidence in the current task.  
 Be willing to move away from the historical average when the task details clearly support it.

At runtime, placeholders were filled per model using leave-one-out statistics over the 93-task calibration set:  $\{N\_held\}$  was typically 92 (93 minus the current task, adjusted for any missing rows for that model). Because SWE-bench Lite tasks cluster by source repository, the optional per-repository line gave the model a more task-relevant prior when enough same-repo data existed.

As a concrete example, the card saved for Claude Opus 4.5 on the `sympy__sympy-17630` instance had  $\{N\_held\}=92$ ,  $\{pass\_rate\}=81.5\%$ ,  $\{mean\_p\}=76.6\%$  (underconfident by 4.9%),  $\{ratio\}=2.4x$ , and the per-repository line covered 21 held-out `sympy/sympy` tasks (pass rate 85.7%, mean stated probability 68.0%).

This card was then followed by the same direct calibration prompt shown in Section 4. The live-scaffold bidding follow-up used the same held-out history idea, but converted it into the stricter hard-prior bid rule described in Section 5.

## B Proof of Proposition 1

*Proof.* Under the three alternative mechanisms, expected welfare is

$$W_H(d) = \lambda_{10}(-c_H) + \lambda_{01}(v - c_H) + \lambda_{11}(v - c_H) + \lambda_{00}(-c_H),$$

$$W_L(d) = \lambda_{10}(v - c_L) + \lambda_{01}(-c_L) + \lambda_{11}(v - c_L) + \lambda_{00}(-c_L),$$

and

$$W_P(d) = (\lambda_{10} + \lambda_{01} + \lambda_{11})v - c_H - c_L.$$

Subtracting from market welfare yields

$$W_M(d) - W_H(d) = \lambda_{10}(v + c_H - c_L) + \lambda_{11}(c_H - c_L) + \lambda_{00}c_H,$$

$$W_M(d) - W_L(d) = \lambda_{01}(v + c_L - c_H) + \lambda_{00}c_L,$$

and

$$W_M(d) - W_P(d) = \lambda_{10}c_H + \lambda_{01}c_L + \lambda_{11}c_H + \lambda_{00}(c_H + c_L).$$

Each expression is weakly positive when  $v > c_H > c_L > 0$ , and it is strictly positive whenever the corresponding alternative makes a mistake with positive probability.  $\square$

## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope?

Answer: [Yes]

Justification: The abstract and introduction state that (i) MarketBench measures whether agents can forecast their own task success and token usage, (ii) six frontier LLMs are miscalibrated on both dimensions, (iii) auctions built from these self-reports diverge from a full-information allocation, and (iv) a self-knowledge prior modestly improves calibration but does not close the gap. Each claim is supported by the empirical results in Sections 4–5.

Guidelines:

- The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: Section 5 discusses run-to-run noise on the 50-task slice (the 6-task gap between two nominally identical market runs), the inability to reject a null of equality between market and solo at  $p = 0.3$ , the live scaffold’s weaker execution environment relative to the external scaffold, and the restriction to a single domain (SWE-bench Lite). Section 6 flags that scalar bids are an inadequate object once budget-quality tradeoffs are first order.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate “Limitations” section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren’t acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

### 3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Proposition 1 states the assumptions ( $v > c_H > c_L > 0$  and the per-task capability indicators  $\pi_i$ ) directly in Section 3. The full proof is in Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

### 4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 4 specifies the 93-task subset of SWE-bench Lite, the six frontier models, the verbatim calibration elicitation prompt, the temperature and JSON-only response format, the breakeven bid formula, the failure penalty, and the reserve-price distribution and number of draws. Section 5 specifies the scaffold's score formula, utility constant, bidding window, execution limit, and two-attempt cap. Appendix A provides the self-knowledge card template and a worked example.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

- (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
- (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
- (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

## 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: SWE-bench Lite is publicly available and cited (Jimenez et al., 2024). The MarketBench task subset, elicitation prompts, model responses, and scaffold/auction simulation code will be released in an anonymized supplementary repository at submission time and as a public repository upon publication.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://neurips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://neurips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Section 4 specifies the 93-task and 50-task slices, the six frontier models, temperature, the JSON response schema, the model-specific blended-pricing rule, and the reserve auction’s failure penalty (\$2) and reserve distribution (Uniform[0,1] with 100 draws and deterministic seeds). Section 5 specifies the scaffold’s utility constant (90 credits), bidding window (90s), execution limit (900s), maximum two attempts per task, and the diagnostic Codex run with an 1800s budget.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Section 5 reports a difference-in-means  $p$ -value of 0.3 for the market-vs-solo pass-rate gap on the 50-task slice and explicitly cautions that single 50-task evaluations are imprecise; the 6-task gap between two nominally identical market reruns is used to characterize the noise floor. Calibration tables report Brier skill relative to a base-rate benchmark. We do not report parametric error bars on per-model auction profits because they are dollar-denominated averages over deterministic reserve draws.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: All experiments use commercial LLM APIs (no local GPU/CPU training). Section 5 reports total token usage for each scaffold condition (e.g., 5.82M tokens for the market run, 4.37M for the solo GPT-5.2 baseline, 321.3M for the `codex-direct` diagnostic) and the per-task wall-clock budget. The auction simulation and analysis scripts run on standard hardware.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

## 9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The research evaluates publicly accessible LLM APIs on publicly available SWE-bench Lite tasks. No human subjects are involved.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

#### 10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Section 6 discusses positive impacts: better self-assessment and market-style coordination could reduce wasted compute and improve allocation across heterogeneous AI agents. Potential negative impacts include the risk that overconfident bids by miscalibrated agents distort allocation in real markets and that operator-run scaffolds could entrench information asymmetries; we flag self-assessment as a training and evaluation target precisely to mitigate these risks.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

#### 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper releases an evaluation benchmark and analysis code, not new models or scraped datasets. The benchmark consists of self-assessment prompts and model responses on publicly available SWE-bench Lite tasks; it poses no notable misuse risk.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

## 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: SWE-bench Lite is cited (Jimenez et al., 2024) and used under its original license. All six evaluated frontier models are identified by name and provider; their APIs were used under their standard terms of service.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](https://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset’s creators.

## 13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: MarketBench (the calibration and auction tasks built atop SWE-bench Lite) is the new asset. The elicitation prompt, response schema, scoring rules, and self-knowledge card template are documented in Section 4 and Appendix A. The release will include the task subset, per-model responses, and scaffold/auction simulation code along with a README and license.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

#### 14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or human subjects. All evaluation is performed against LLM APIs.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

#### 15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

#### 16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs are the primary object of study. The six frontier models are identified by name in Section 4 (Claude Opus 4.5, Claude Sonnet 4.5, Gemini 3 Pro Preview, GPT-5.2, GPT-5.2-pro, GPT-5-mini). All elicitation prompts, the breakeven bid formula, and the scaffold scoring rule that consume LLM outputs are described in the main text.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.