REALFIN: Benchmarking Agent Skills for Multi-Tool Orchestration under Tool Explosion

Anonymous ACL submission

Abstract

Large language models are increasingly moving from answering questions to acting as agents that plan actions, call tools and operate in external environments. Solving tasks in these settings requires constructing and executing multi step workflows rather than producing a single response. The key difficulty lies not only in computation but in choosing appropriate tools and organizing them into effective workflows, since realistic environments contain many tools with related yet non identical functions and several plausible solution paths may coexist for the same task. However, many existing benchmarks primarily emphasize static question answering and final outputs, often using simplified tool environments and short workflows, which provides limited visibility into how tools are chosen and workflows are executed. A natural direction is therefore to evaluate agents through the processes they carry out in realistic environments where overlapping tools and genuine workflows are present. We introduce REALFIN to support this direction. REALFIN contains tasks derived from authentic financial analysis workflows, pairs each task with an executable reference program and operates in realistic multi tool environments while recording execution traces. It enables joint evaluation of outcomes and execution behavior and supports process oriented assessment of tool choice, workflow organization and temporal grounding in addition to answer accuracy.

1 Introduction

Large language models are increasingly moving from answering questions to acting as agents. Modern systems do not only produce textual outputs but plan actions, call tools and operate within external environments. In these settings, obtaining an answer requires constructing and executing multi step workflows rather than performing a single step generation. This shift brings evaluation closer to the way real tasks are completed in practice.

The key difficulty in this agent setting lies in choosing and organizing tools rather than in computation alone. Realistic environments contain many tools whose functions are related yet not identical, and several distinct workflows may appear reasonable for the same task. Multiple solution paths therefore coexist and the agent must decide which tools to invoke and how to arrange them. This phenomenon, in which overlapping tools create abundant and plausible workflows, can be viewed as tool explosion and it places central importance on agent skills in tool selection and workflow orchestration.

However, existing benchmarks do not primarily target this form of agent behavior. Many widely used evaluations are organized as static question answering (Chen et al., 2021, 2022; Zhu et al., 2021; Deng et al., 2022; Guo et al., 2025) tasks that emphasize final answers, providing limited visibility into how tools are chosen and workflows are executed. In a number of settings the tool environments are simplified (Qin et al., 2024a; Liu et al., 2024) and the workflows required to complete tasks are short. These benchmarks serve important purposes but focus on different aspects of capability from those that arise when rich tool ecosystems and extended workflows are naturally present.

A natural direction is therefore to evaluate agents through the processes they execute in realistic environments. When tools overlap in function and workflows are required to reach answers, the behavior of interest is the way agents plan, choose tools, ground temporal references and coordinate intermediate steps. Evaluations that observe such processes complement answer accuracy and offer a more complete view of the abilities that underlie reliable tool use.

We propose REALFIN from this perspective. REALFIN consists of tasks constructed from authentic

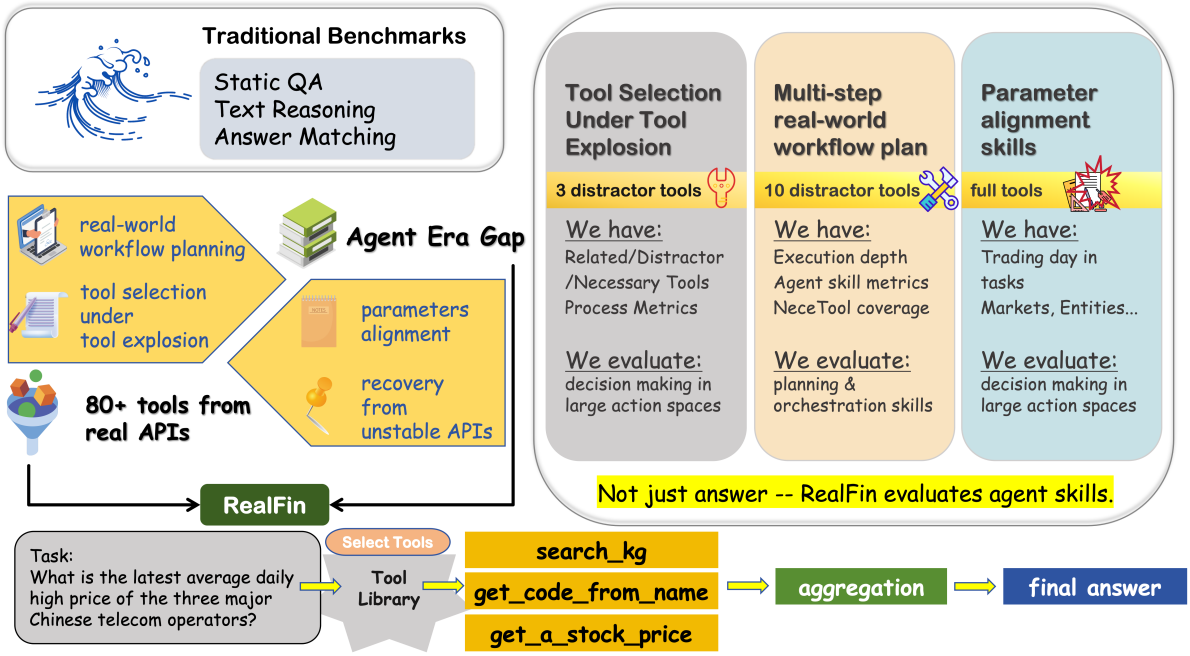

Figure 1: Overall concept of REALFIN: workflow-grounded financial tasks, executable golden code, and a realistic multi-tool environment with tool explosion.

financial analysis workflows that require multi step tool use. Each task is paired with an executable reference program that specifies one valid solution path while still permitting alternative workflows in practice. The benchmark is built on realistic financial application programming interfaces and records execution traces, enabling the study of both outcomes and processes in environments where tool abundance arises naturally.

Our contributions are threefold.

- We advance a workflow-grounded benchmark for financial agents, framing tasks as complete analysis workflows rather than static QA, instantiated in REALFIN.
- We show that realistic financial APIs naturally give rise to tool explosion, and we instantiate this phenomenon in a live multi-tool environment with redundancy and overlapping functionality.
- We promote a process-based view of agent evaluation, where models are assessed through their tool-use behavior rather than only final answers, and REALFIN makes this view measurable under tool explosion.

2 Related Work

2.1 Financial reasoning benchmarks

Existing financial benchmarks mainly evaluate static question answering and numerical reasoning rather than executable workflows. Representative datasets such as FinQA, ConvFinQA, TAT-QA and FinEval (Chen et al., 2021, 2022; Zhu et al., 2021; Guo et al., 2025) are built on fixed corpora with a single golden answer and do not require real API interaction. They therefore focus on knowledge and reasoning rather than tool use, workflow execution, or agent behavior in live financial systems.

2.2 General tool-use and agent benchmarks

General agent benchmarks study tool use and environment interaction through multi-tool or web-based tasks (Qin et al., 2024b; Li et al., 2023; Zhou et al., 2024; Liu et al., 2024; Mialon et al., 2024). Many employ simulated or mirrored APIs for reproducibility, while others connect to real systems but offer limited diagnostic control. Tool explosion usually appears only as a byproduct rather than a primary focus, whereas our work explicitly targets agent behavior in redundant multi-tool environments with executable workflows.

Benchmark	Tool Selection	Multi-Tool Workflow	Real-API Robust-ness	Temporal Reasoning	Alt. Tool Paths	Tool Propen-sity
FinQA	✗	✗	✗	✗	✗	✗
ConvFinQA	✗	✗	✗	✗	✗	✗
FinanceBench	✗	✗	✗	△	✗	✗
FinBen	△	△	✗	△	✗	✗
FinSearchComp	✓	△	△	✓	✗	△
ToolBench	✓	✓	△	✗	✗	✓
GAIA	✓	△	△	△	✗	△
BFCL	✓	△	△	✗	✗	✓
RealFin (Ours)	✓	✓	✓	✓	✓	✓

Table 1: Benchmark coverage across capability dimensions. Symbols: ✓ = supported, ✗ = not supported, △ = partially or indirectly supported. Dimensions: Tool Selection = choosing correct tools from large catalogs; Multi-Tool Workflow = composing multi-step domain workflows; Real-API Robustness = robustness to unstable real-world APIs and execution noise; Temporal Reasoning = grounding to trading calendars and temporal anchors; Alt. Tool Paths = support for multiple valid executable tool workflows; Tool Propensity = tendency to invoke tools instead of zero-tool answering.

3 Benchmark Design

3.1 Workflow-Driven Task Construction

Workflow grounded task design We construct REALFIN directly from full financial analysis workflows used by human analysts. Tasks are decomposed into core analytical intents intermediate variables and dependency structures. Each instance is aligned with an executable solution path so the benchmark evaluates planning and execution under tool explosion instead of single step answering.

Template based large scale task generation Workflow skeletons are transformed into parameterized templates and instantiated under financial validity constraints. Tasks are executed in a real API environment that preserves overlapping and substitutable tools so multiple valid tool chains may solve the same task.

Reflecting tool explosion through a redundant multi tool environment We construct a multi tool execution environment from real financial APIs (Akshare (King, 2022)) that intentionally preserves functional overlap and substitutable tools. Tasks are not bound to a single correct tool. The golden workflow specifies one valid solution path while alternative paths may also exist. This embeds tasks in a naturally redundant tool space so that tool explosion arises from the environment itself rather than from artificial task design.

Executable alignment of tasks and solutions Every task in REALFIN is paired with an executable golden program. The program encodes the full solution process including entity resolution tool selection parameter planning temporal alignment and aggregation. Programs are executed against real APIs and samples that are non executable unstable or ambiguous are removed. This strict alignment enables process level evaluation instead of answer only scoring.

3.2 Dataset Structure and Properties

This section overviews the structural properties of REALFIN, including its data fields, tool specifications, task distributions, and comparison with representative benchmarks. We aim to clarify what REALFIN is and why it is suitable for evaluating agent skills, particularly workflow execution under tool explosion. Figure 2 illustrates the joint design of the data structure and the tool structure in REALFIN.

Temporal structure Tasks are categorized by temporal behavior into Fixed, Relative, and Real-time classes. This induces trading-calendar alignment, rolling windows, and data drift, making temporal grounding an evaluated agent skill rather than mere post-processing.

Table 2: Benchmark comparison. RealFin denotes the full benchmark, while RealFin-Hard denotes the subset of tasks whose reference workflows require at least five tool calls.

Benchmark	Domain	#Tasks	#Tools/APIs	Avg Tool Calls	Avg Exec Depth	Exec	Time-Sens.	Answer type
RealFin	Finance (tool)	1524	85	3.78	2.12	✓	✓	Fast Live exec
RealFin-Deep	Finance (tool)	482	85	6.10	2.10	✓	✓	Live exec
API-Bank	Tool-aug LLM	274	53	2.07	≈1.50	✓	×	Simu APIs
BFCL	General	5551	/	≈2.55	≈1.50	✓	×	AST
ToolBench	General	126,486	16,464	3.71	≈2.00	✓	×	Static
PACIFIC	Fin conv-QA	2757	Document	0.00	≈1.00	×	×	Static
WebArena	Web agent	812	Web	≈2.8	≈2.80	✓	✓	Rule-based
ConvFinQA	Fin num-reason	14115	computation	≈1.00	≈1.50	✓	×	Static
FinQA	Fin num-reason	8281	computation	≈2.00	≈1.50	✓	×	Static
FinSearchComp	Fin QA	635	search engine	1–2	1–2	×	✓	LLM-Judge

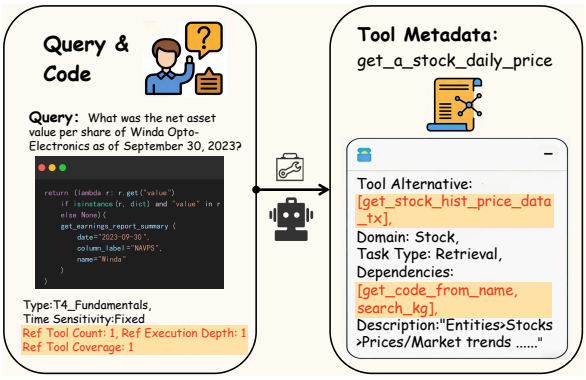

Figure 2: Data and tool structure in REALFIN. The left side illustrates the task-side representation including query text and executable code, while the right side shows the structured tool metadata such as domain, task type, dependencies, and alternatives.

Tools and multi-path structure REALFIN contains about 85 overlapping real financial tools, yielding a shallow hub-dominated dependency graph and multiple valid tool chains per task. Tool explosion therefore arises naturally and agents must select, route, and orchestrate tools instead of only producing final answers.

Table 2 shows that REALFIN uniquely combines real APIs temporal sensitivity multi tool redundancy and executable workflows making it suitable for evaluating agent skills under tool explosion.

We further validate that golden workflows align with real analyst practice by comparing them with standard operating procedures. Representative cases are given in Appendix D.

3.3 Quality Control and Evaluation Protocol

We implement a quality-control and evaluation protocol that secures reliable dataset construction, executable workflows, and reproducible and interpretable evaluation outcomes.

Data quality control We apply structured validity checks to all template instantiated tasks, filtering nonexistent or ambiguous entities, invalid market codes, non trading days, and insufficient historical windows. Each task is paired with a golden reference program and executed against real financial APIs. Instances that fail or return empty results are repaired or removed. Golden programs are repeatedly rerun to test output consistency, using rule based checks, selective manual inspection, and periodic re execution to monitor temporal drift and to separate market movement from API noise. Thus every sample is semantically valid and end to end executable.

Golden programs are end-to-end executable in practice. Execution success exceeds 97% and repeated runs show high output stability, while only a very small fraction of temporally labeled tasks change across dates, reflecting genuine market non-stationarity.

Evaluation protocol We adopt a process aware evaluation protocol rather than answer only scoring. In addition to outcome accuracy we compute process metrics from full execution traces including entity binding temporal grounding tool selection routing order parameter planning and multi step orchestration. These metrics enable diagnosis of specific agent skills and failure modes. Because real APIs may fail we explicitly separate system errors from model errors by reporting metrics on all traces and on the subset in which every API call succeeds. REALFIN supports both snapshot and live evaluation. Snapshot mode freezes answers for reproducibility. Live mode preserves non stationarity for realism. Both yield consistent qualitative conclusions.

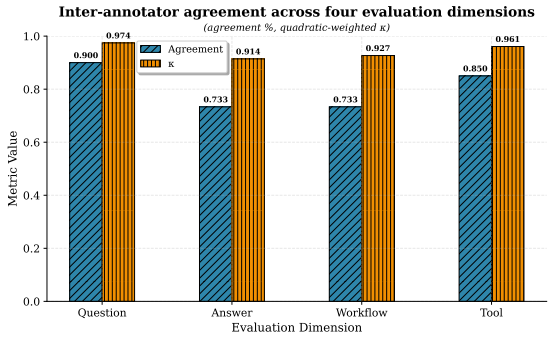

Figure 3: Inter annotator agreement across four evaluation dimensions

3.4 Human validation of data quality

We conduct a human validation study to assess the reliability of questions answers workflows and tools in the benchmark. The questionnaire is in Appendix F.

Setup We uniformly sample 60 tasks spanning diverse workflow depths and task types. Each task is independently reviewed by two annotators, yielding 120 double-annotated instances. We recruit 12 annotators with backgrounds in quantitative finance and computer science, including graduate students and industry practitioners. Tasks are evaluated on four dimensions—question well-posedness, answer correctness, workflow correctness, and tool usability—using an ordinal 0–1–2 scale. We report raw agreement and quadratic weighted κ based on all double annotations.

Results and interpretation Figure 3 show that raw agreement ranges from 0.73 to 0.90 and quadratic weighted κ reaches 0.91–0.97 across all four dimensions, indicating near perfect agreement. Most disagreements are between 1 and 2 rather than between 0 and 2, meaning annotators mainly differ in strictness rather than in judging validity. These results confirm that questions are well posed, answers are correct, workflows are executable, and tools are usable.

4 Experiments

This section evaluates whether large language models possess the agent skills needed to complete financial workflows under tool explosion. Agent skills include tool selection in redundant tool sets, multi-step workflow orchestration, semantic grounding of entities and time, parameter planning, and error recovery. We vary tool exposure (Full, BM25, Oracle, OraA-3, OraA-10), add

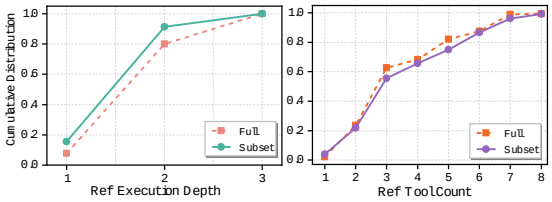

Figure 4: **Structural similarity between the full benchmark and the 128-core subset.**

structured assistance (package routing and workflow hints), and analyze process-level metrics and tool traces. Results show systematic deficiencies in these skills within large and functionally overlapping tool ecosystems.

4.1 Experimental Setup

Evaluation subsets. We evaluate on a 128-task core subset. Its execution-depth and tool-count distributions match the full benchmark (Figure 4), with no significant differences by Kolmogorov–Smirnov tests ($p = 0.088, 0.548$). The subset therefore serves as a compact but faithful representation of the full dataset.

(1) Baselines. *Full* exposes all tools to the model, simulating real systems and the strongest level of tool explosion. *BM25* exposes only the Top-20 tools retrieved by BM25, simulating retrieval-based tool selection and testing whether shallow tool discovery mitigates tool explosion.

(2) Controlled exposure. *(i) Oracle (Ora).* Expose only tools used in the reference workflow, largely removing tool explosion and isolating execution, parameter planning, and temporal grounding. *(ii) OraA-k.* Extend Oracle with functionally similar alternatives and k distractors, increasing tool redundancy while keeping task semantics fixed. We report OraA-3 and OraA-10.

(3) Package routing. *Pkg* groups tools by functionality; the model first selects a package and then a tool within it. This provides limited structural guidance and examines whether hierarchical routing alleviates difficulty in large tool spaces.

(4) Unfair oracle-guided setting. In this unfair oracle-guided setting, the model answers with oracle trajectory hints including the correct tool set and solution path but not the final answer.

4.2 Metrics

We report both outcome-level and process-level metrics. Outcome metrics measure whether a task is successfully completed. Process metrics are de-

Table 3: Evaluation metrics.

Metric (Abbrev.)	Definition
Outcome-level metrics	
Outcome Accuracy (Acc)	Answer-only.
Execution Success Rate (ESR)	Calls succeed.
Process-level metrics	
Tool Selection Precision (TSP)	Correct tools used.
Tool Selection Recall (TSR)	Correct tools succeed.
Ordering Accuracy (OA)	Call order right.
Entity Grounding Accuracy (EGA)	Entities correct.
Temporal Grounding Accuracy (TGA)	Time correct.
Parameter Planning Success (PPS)	Parameters correct.
Orchestration Success Rate (OSR)	Workflow correct.
Recovery from Error (RFE)	Errors fixed.

rived from tool-use traces and characterize agent behavior under tool explosion beyond final-answer correctness. (Table 3)

4.3 Main results

Table 4 reports overall performance on the REALFIN-128 core subset. The agent skills considered here include tool discovery and selection, multi-step workflow orchestration, parameter planning, entity and temporal grounding, and error recovery.

Existence of tool explosion. In our setting, tool explosion refers not merely to larger tool sets, but to the coexistence of multiple functionally related tools for the same sub-task that must be composed in multi-step workflows, inducing a rapidly expanding choice space. Table 4 shows that from Oracle to OraA-10, ACC drops from about 20–35% to 5–15%, and OSR drops from about 5–15% to nearly zero for most models. This consistent decline under constant tasks provides direct empirical evidence for tool explosion.

Controlled manipulation of distractor intensity. Keeping the task and reference workflow fixed, we progressively increase both functionally related tools and distractor tools. Figure 6 reports ACC, OSR, and PPS under Oracle, OraA-3, OraA-10, Full, and BM25 settings. As the tool space grows, all three metrics decrease systematically, with a clear threshold effect: changes from Oracle to OraA-3 are moderate, while the drop from OraA-3 to OraA-10 and Full is much sharper. This indicates that the behavioral changes are driven by the structure of the tool space rather than by differences in task difficulty or data noise.

Systematic degradation of agent skills under tool explosion. In the Full setting, ACC typically lies around 5–15%, while OSR is often near zero. PPS and RSS also remain low, usually below 20%. These results indicate that degradation reflects a systematic decline of multiple agent skills, including tool selection, workflow orchestration, parameter planning, and recovery, under tool explosion.

Association between tool-space size and performance degradation. To separate task difficulty from tool-space effects, we reduce the candidate pool using BM25 retrieval. Compared with Full, both ACC and OSR increase by roughly 2–4 times, with PPS likewise improving. This consistent trend across models indicates a stable quantitative relationship between performance degradation and tool-space expansion.

Effect of structured interface organization. The package-routing setting keeps the tool count unchanged, but groups tools functionally and enforces hierarchical selection. ACC and OSR are consistently higher than in Full, with OSR increasing from near zero to 10–20%, and PPS improving correspondingly. This shows that structured interfaces can mitigate tool selection and orchestration burden without reducing tool quantity.

Answer correctness does not imply workflow completion. ACC and OSR are often decoupled. ACC can reach 20–40% whereas OSR remains in single digits or near zero. Thus, many correct answers do not arise from correct end-to-end execution but from guessing or partial use of tools. Relying solely on answer-level metrics therefore overestimates true orchestration and recovery ability, underscoring the necessity of process-level evaluation.

Task solvability and locus of difficulty. With oracle trajectory hints, models achieve 83% accuracy, showing that tasks are solvable and that most difficulty lies not in task ambiguity but in autonomous tool discovery and workflow planning.

Variance analysis. Task-level bootstrap with exact binomial intervals (Table 16, Appendix E) shows that observed differences cannot be explained by sampling noise. We identify 19 large effects ($|\Delta| \geq 0.05$, $|z| \geq 2$) with mostly non-overlapping confidence intervals. Execution metrics (`exec_any`, `exec_oracle`) drop by 30–76%, while tool-selection precision remains stable (>0.9). This indicates that degradation under tool explosion is systematic rather than variance noise.

Table 4: Model performance across tool-exposure regimes. ACC = outcome accuracy; OSR = orchestration success; PPS = parameter-planning success; RSS = recovery success.

	Baseline								Constrained tool spaces												Package-route			
Model	Full*				BM25*				Oracle				OraA-3				OraA-10				pkg			
	ACC	OSR	PPS	RSS	ACC	OSR	PPS	RSS	ACC	OSR	PPS	RSS	ACC	OSR	PPS	RSS	ACC	OSR	PPS	RSS	ACC	OSR	PPS	RSS
Cluster A																								
Claude	9.4	0.0	0.8	8.5	25.0	14.1	29.7	36.6	27.3	3.1	13.3	9.9	36.7	14.1	48.4	17.9	26.6	1.6	13.3	9.8	23.4	17.2	18.0	8.3
GPT	1.6	0.0	0.8	9.4	6.4	4.7	11.7	26.0	10.9	2.3	7.8	5.8	8.6	11.7	35.2	15.4	4.3	0.0	2.9	11.1	30.5	15.6	18.0	6.6
Gemini	8.6	0.0	0.8	8.5	10.0	5.5	7.8	13.8	14.8	2.3	12.5	9.1	9.4	0.8	9.4	11.4	7.0	1.6	1.6	4.9	38.3	12.5	17.2	10.7
Cluster B																								
Qwen	3.1	0.0	2.3	6.7	9.6	2.3	6.2	15.4	16.4	1.6	14.1	9.1	14.1	10.9	35.2	13.8	4.7	0.0	3.1	5.7	23.4	13.3	14.8	10.7
DeepSeek	30.5	0.0	8.6	11.6	9.4	18.0	26.6	35.8	32.0	5.5	13.3	8.3	3.9	0.0	0.0	36.4	0.8	0.0	0.0	30.2	19.5	13.3	14.1	9.9
Doubao	33.6	0.8	16.4	6.6	11.7	7.8	23.4	37.4	36.7	4.7	19.5	9.1	0.0	0.0	0.0	31.4	0.0	0.0	0.0	26.4	42.2	11.7	14.8	11.6
Kimi	10.2	0.8	7.8	7.4	4.7	13.3	21.1	30.9	30.5	0.8	14.1	8.3	1.6	0.0	0.0	17.4	0.0	0.0	0.0	15.7	14.1	2.3	4.7	9.1
Unfair oracle-guided setting's ACC $\approx$ 83.59%																								

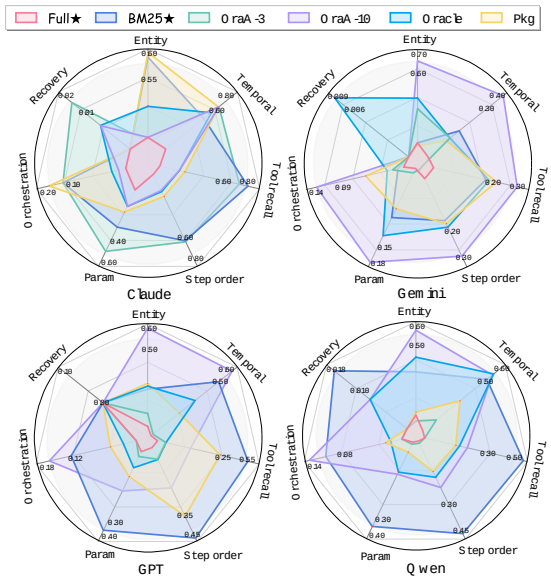

Figure 5: Agent-skill radar visualization across seven process metrics. Outer envelopes correspond mainly to OraA-3 and coarse-pkg settings, while Full and OraA-10 settings contract sharply toward the center.

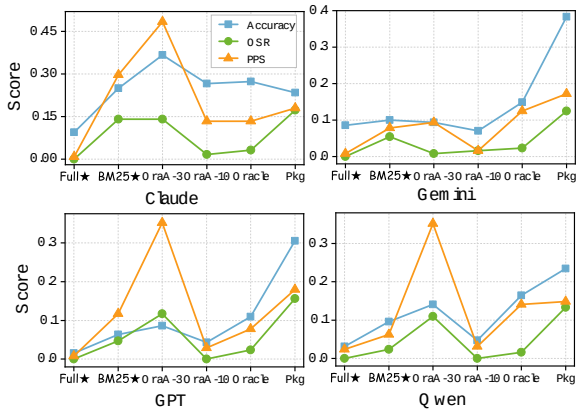

Figure 6: **Effect of distractor intensity.** ACC, OSR, and PPS under Oracle, OraA-3, OraA-10, Full, and BM25 on the 128-core subset. Performance peaks at OraA-3 and drops sharply as distractors increase.

4.4 Agent-skill-oriented analysis

To move beyond the limitations of answer-level metrics, we further analyze agent behavior in REALFIN using process-based measures. The radar plot in Figure 5 summarizes seven dimensions, including tool recall, step consistency, parameter planning, entity and temporal grounding, workflow execution success, and error recovery, which together characterize the core agent skills required to complete executable financial workflows.

Tool selection as a root capability Tool recall and step consistency move closely together across exposure settings. When tool recall declines, step consistency almost always collapses as well. This pattern indicates that many ordering errors arise after the wrong tools have been selected, rather than from step planning alone. Tool selection therefore functions as a root capability for workflow construction. Once tool explosion enlarges the action space and weakens this capability, failures then propagate to ordering and orchestration.

Limited replanning and recovery. Error recovery remains low across settings and deteriorates further as the tool space expands. Execution traces show that once models commit to an incorrect tool or path early on, they rarely abandon it and replan, but instead continue to iterate along the wrong route. This behavior suggests that current models behave more like one-shot planners with little genuine reflection and replanning, and are therefore easily locked into failure under tool explosion.

Process skills beyond answer accuracy. Table 17 in Appendix E evaluates whether REALFIN measures process skills rather than only answer accuracy. Retrieval failure is the largest error source, indicating that identifying the correct tools in a

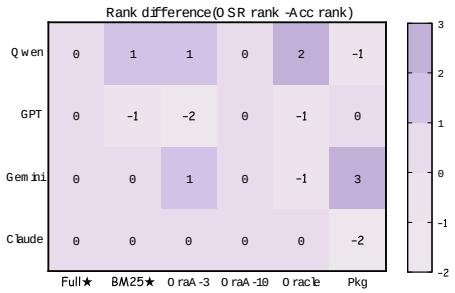

Figure 7: **Rank–flip heatmap.** Difference between OSR-based and ACC-based model rankings across exposure regimes. Large flips occur in moderate tool spaces; very large spaces flatten OSR.

large action space is already difficult. Even when the right tools are retrieved, many cases still fail due to ordering, parameter choice, or temporal grounding.

Answering and executing induce different model rankings. We compare model rankings induced by ACC and OSR. Figure 7 shows that the two rankings diverge most under medium tool-space sizes such as Oracle and OraA-3. In these settings, models may answer correctly without finishing the workflow, or complete the workflow but still miss the correct final value. Under very large tool spaces such as OraA-10 and Full, OSR collapses toward zero and execution-based rankings become largely flattened. Table 17 further shows that most failures occur at the tool selection stage, while $\mathrm{ACC} \mid \text{selection} \gg \mathrm{ACC}$, indicating that the main bottleneck lies in tool selection and workflow orchestration rather than single-step tool invocation.

Shaping effects of interface structure Interface organization markedly shapes agent behavior. Package routing and retrieval-based exposure, although not reducing the number of tools, stabilize tool selection and improve orchestration success by imposing external structure on decisions. These gains mainly reflect reduced decision burden rather than stronger intrinsic skills, as recovery and replanning improve only marginally. Qualitative cases illustrating these effects are shown in Appendix C.

4.5 Analysis under controlled interventions

Tool explosion enlarges the agent's action space. Rather than making causal claims, we conduct two **quasi-experimental** interventions to examine how (i) tool-space size and (ii) context length **relate to** agent behaviour.

Intervention 1 tool-space size We fix the prompt context and vary only the number of available tools. We evaluate $k \in \{0, 3, 6, 10\}$ distractors under the G+R+kN configuration while keeping tasks and prompts identical. Qwen is reported as a representative model because it avoids ceiling and floor effects and shows clear variation in ACC, TSR and OSR across k (Table 19).

As k increases, TSR and OSR monotonically decrease and OSR is essentially zero for $k \geq 6$. This pattern is consistent with enlarged tool spaces making tool selection and orchestration more difficult, although changing k may also influence other prompt properties.

Intervention 2 context length We then fix the tool set (OraA-3) and vary only prompt length using short and long versions of the same tasks on a 30-task subset. Both Claude and Qwen are evaluated. Despite different absolute scores, both models show the same qualitative trend (Table 18).

Longer contexts are consistently associated with lower ACC, TSR and PPS, and OSR drops to zero for both models, indicating degraded tool selection and orchestration even when the action space is fixed.

Further quantitative and conditional error analyses are provided in Appendix E.1.

5 Conclusion

We presented REALFIN, a benchmark for evaluating LLM agents in realistic financial workflows. It emphasizes agent skills rather than answer matching, including tool selection, multi-step orchestration, parameter planning, temporal grounding, and error recovery. Each task is paired with an executable reference program and interacts with real financial APIs, enabling process-level evaluation.

Experiments under controlled tool exposure demonstrate tool explosion effects: adding related tools while fixing task semantics consistently degrades accuracy and orchestration success. The main performance drop concentrates on tool selection and cascades to execution, and answer-based and process-based metrics yield different model rankings. With workflow guidance, accuracy increases markedly, showing that tasks are solvable and the difficulty mainly lies in agent skills.

REALFIN serves as a diagnostic benchmark, offering an executable and process-aware testbed for studying agent behavior in dense tool ecosystems and time-sensitive environments.

Limitations

Although REALFIN is designed to characterize tool use and workflow execution in real financial environments, the benchmark has several limitations. First, its construction is constrained by the financial tool ecosystem and market coverage currently supported, and it does not exhaust all financial subdomains or data-source types. Model behavior under different market institutions or tool systems may therefore exhibit additional challenges. Second, despite incorporating temporal sensitivity and real APIs, our tasks and workflows are still abstracted and parameterized, and REALFIN covers only a representative subspace of the fully open-world setting. Third, the quality-control and evaluation protocols aim to separate system noise from model behavior, but inevitably depend on implementation details and runtime environments, motivating future studies on cross-platform and cross–data-source consistency.

Overall, these limitations do not undermine the role of REALFIN in studying agent skills in realistic tool environments. Instead, they point to future extensions, including broader market coverage, richer tool types, increased task openness, and transferable evaluation across ecosystems.

Use of AI assistants. During the preparation of this paper, we used large language models (including ChatGPT) to assist with tasks. All technical ideas, dataset design, experiments, and conclusions were conceived and verified by the authors. All generated text was checked and edited by the authors, and we take full responsibility for the content of the paper.

Ethics Statement

REALFIN is an evaluation benchmark rather than a deployed decision-making system. It is designed for scientific study of agent behaviour and does not provide investment advice or trading signals. Nevertheless, evaluating agents in financial settings raises several ethical considerations.

First, we clarify the provenance of the data used in REALFIN. All tasks are generated from publicly accessible financial data sources and public API interfaces. The benchmark does not contain personal data, non-public market information, or proprietary client records. No human subjects were involved, and no manual annotations of personal content were collected.

Second, we consider the risk of potential misuse. Although REALFIN focuses on evaluation rather than training or deployment, stronger agentic capabilities in finance could in principle be used for harmful purposes, including market manipulation, automated speculation, or amplified misinformation about financial assets. To mitigate these risks, the benchmark provides no trading APIs and does not include tasks that directly optimise or recommend financial actions. The benchmark is intended for controlled research environments rather than end-user decision automation.

Third, we reflect on possible systemic effects. Real financial APIs and time-varying data reflect real-world conditions, but agents evaluated on REALFIN should not be assumed to be safe or reliable for high-stakes use. We discourage interpreting benchmark scores as certification for deployment in financial decision-making pipelines; rigorous domain-specific oversight remains necessary.

Fourth, we discuss fairness and accessibility. The benchmark reflects financial markets and APIs that are available in our environment and may not represent global financial practices equally. We release detailed documentation and open-source code to support transparency and reproducibility, while acknowledging that API access may depend on regional availability.

Overall, we emphasize that REALFIN aims to support responsible research into agent evaluation by making processes observable and failure modes explicit, rather than by enabling autonomous financial decision systems. Researchers using this benchmark should consider the broader societal impact of agentic AI in finance and avoid applications that could lead to individual or systemic harm.

References

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Kenneth Huang, Bryan R. Routledge, and William Yang Wang. 2021. Finqa: A dataset of numerical reasoning over financial data. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pages 3697–3711. Association for Computational Linguistics.

Zhiyu Chen, Shiyang Li, Charese Smiley, Zhiqiang Ma, Sameena Shah, and William Yang Wang. 2022. Convfinqa: Exploring the chain of numerical reasoning in conversational finance question answering. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United

Arab Emirates, December 7-11, 2022, pages 6279–6292. Association for Computational Linguistics.

Yang Deng, Wenqiang Lei, Wenxuan Zhang, Wai Lam, and Tat-Seng Chua. 2022. PACIFIC: towards proactive conversational question answering over tabular and textual data in finance. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022, pages 6970–6984. Association for Computational Linguistics.

Xin Guo, Haotian Xia, Zhaowei Liu, Hanyang Cao, Zhi Yang, Zhiqiang Liu, Sizhe Wang, Jinyi Niu, Chuqi Wang, Yanhui Wang, Xiaolong Liang, Xiaoming Huang, Bing Zhu, Zhongyu Wei, Yun Chen, Weining Shen, and Liwen Zhang. 2025. Fineval: A chinese financial domain knowledge evaluation benchmark for large language models. In Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025, pages 6258–6292. Association for Computational Linguistics.

Albert King. 2022. Akshare. https://github.com/akfamily/akshare.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. Api-bank: A comprehensive benchmark for tool-augmented llms. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023, pages 3102–3116. Association for Computational Linguistics.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, and 3 others. 2024. Agentbench: Evaluating llms as agents. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2024. GAIA: a benchmark for general AI assistants. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024a. Toolllm: Facilitating large language models to master 16000+ real-world apis. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024b. Toolllm: Facilitating large language models to master 16000+ real-world apis. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. Webarena: A realistic web environment for building autonomous agents. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021, pages 3277–3287. Association for Computational Linguistics.

Appendix

A Dataset Statistics

Type distribution. Table 5 summarizes the distribution of task types in REALFIN.

Type	Count	Proportion
T3_Aggregate	677	44.4%
T1_Snapshot	412	27.0%
T4_Fundamentals	328	21.5%
T2_Compare	107	7.0%

Table 5: Distribution of task types in the 1,524-instance dataset.

Takeaways.

- 1,524 valid instances, no broken records.
- Aggregate questions (T3) constitute nearly half of the corpus.

Execution depth.

Tool usage statistics. **Takeaways.**

- 567 single-tool vs. 957 multi-tool instances (37.2% vs. 62.8%).
- 433 instances (28.4%) require actual chaining (depth ≥ 2).

Time Sensitivity	Count	Proportion
Fixed	794	52.1%
Relative	629	41.3%
Realtime	101	6.6%

Table 6: Time-sensitivity labels for dataset instances.

Time sensitivity. **Type $\times$ Time Sensitivity.** **Depth $\times$ Time Sensitivity.** **Takeaways.**

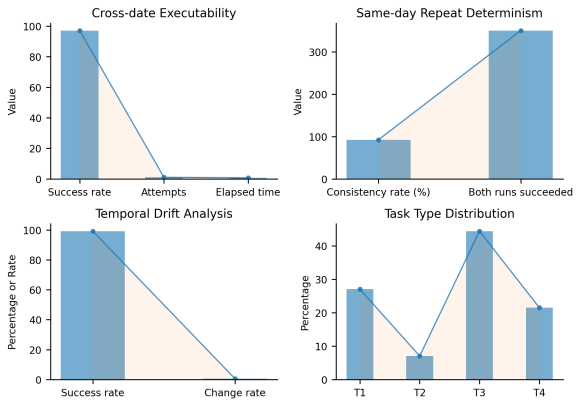

Figure 8: **Code analysis experiments.** We evaluate executability across dates, determinism across repeated runs, and temporal drift of time-sensitive tasks. High executability with non-trivial drift confirms that REALFIN captures realistic, time-varying financial environments.

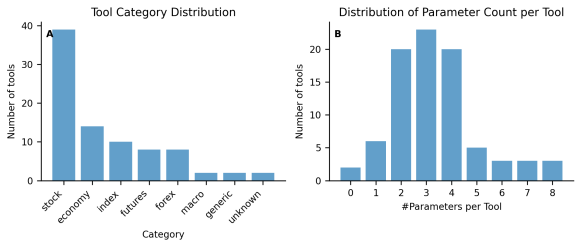

Figure 9: **Tool ecosystem statistics.** Left: distribution of tool categories. Right: distribution of parameter counts per tool. The benchmark exhibits heterogeneous and partially redundant tools, operationalizing the challenge of tool explosion.

- Fixed-date queries dominate; relative-time queries are substantial.
- Realtime queries are mostly shallow and largely in T1.
- Deeper chains skew toward relative-time computation.

Workflow complexity signals. **Takeaways.**

- Control flow is pervasive despite shallow average depth.
- Multi-tool usage is common; long chains are less frequent.

Extended Dataset Statistics.

B Tool Library Statistics

Domain distribution. **Takeaways.**

- Stock-related tools dominate the library.

Type	Fixed	Realtime	Relative
T1_Snapshot	315	97	0
T2_Compare	102	4	1
T3_Aggregate	49	0	628
T4_Fundamentals	328	0	0

Table 7: Joint distribution of task type and time sensitivity.

Signal	Count	Proportion
Control flow in code	1436	94.2%
Multiple tools (>1)	957	62.8%
Chained tools (depth $\geq$ 2)	433	28.4%

Table 8: Indicators of workflow complexity.

- Long tail of mixed-domain tools reflects cross-asset workflows.

Task types of tools. **Takeaways.**

- The library is primarily retrieval-oriented; computational tools are secondary.

Dependency graph properties. **Graph statistics.**

- Leaves: 46
- Internal nodes: 39
- Longest chain length: 4

Hub tools (fan-in).

- `get_code_from_name`: 18
- `search_kg`: 4
- `get_index_history`: 3

Fan-out.

- `calculate_gold_forex_change`: 2
- `get_a_stock_daily_price`: 2

Takeaways.

- The dependency graph is shallow with a few hub tools.

Error models. **Takeaways.**

- 97.6% of tools expose structured error models.
- Parameter and internal errors dominate.

Statistic	Value
Number of questions	1,524
Executable golden codes	1,524
Tool functions	85
Markets covered	A, HK, US, FX, Macro
Task types (T1/T2/T3/T4)	27% / 18% / 44% / 11%
Avg. execution depth	2.12
Avg. tool calls per instance	3.78
Multi-tool questions	73%
Temporal regime: Fixed	28.3%
Temporal regime: Relative	52.1%
Temporal regime: Realtime	19.6%
Golden program success rate	97.15%
Avg. attempts per program	1.06
Repeat-run consistency	87.64%
Temporal drift proportion	0.55%

Table 9: **Statistics of REALFIN.** All questions are paired with executable golden programs. Executable analysis is based on automatically re-running all golden programs across dates.

Domain	Count	Proportion
stock	37	43.5%
index	10	11.8%
economy	10	11.8%
futures	7	8.2%
forex	7	8.2%
generic	4	4.8%
mixed domains	8	–

Table 10: Domain coverage of tools. Mixed-domain rows aggregate tools annotated with multiple domains.

Real APIs vs. virtual tools. Takeaways.

- Rough balance between wrapper tools and virtual/algorithmic tools.
- Akshare is the dominant external data provider.

C Full metric definitions

This appendix provides full mathematical definitions of the metrics introduced in Section §4. We distinguish between outcome accuracy and process-based agent-skill metrics.

C.1 Notation

Let the dataset be $\mathcal{D} = \{1, \ldots, N\}$. For each instance $i \in \mathcal{D}$:

- y_i : reference answer
- $\hat{y}_i$: model's final answer
- $T_i^* = \langle t_{i,1}^*, \ldots, t_{i,m_i}^* \rangle$: gold tool sequence
- $\hat{T}_i = \langle \hat{t}_{i,1}, \ldots, \hat{t}_{i,k_i} \rangle$: model tool sequence

Metric	Value
Total tools	133
External-API tools	65
Virtual / algorithmic tools	68
Unique Akshare endpoints	66

Table 11: Real vs. virtual tool implementation statistics.

- $\mathcal{T}_i^*$: set of gold tools (ignoring order)
- $\hat{\mathcal{T}}_i$: set of tools used by the model
- d_i^* : gold workflow depth $= |T_i^*|$
- $\hat{d}_i$: model workflow depth $= |\hat{T}_i|$
- $r_{i,j}$: execution result of tool call $\hat{t}_{i,j}$

We write $\mathbf{1}[\cdot]$ for the indicator function.

C.2 Outcome-Level Metrics

Outcome Accuracy (Acc).

$$\text{Acc} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[\hat{y}_i = y_i]. \tag{1}$$

C.3 Execution and Orchestration

Execution Success Rate (ESR). Let $\text{OK}(r)$ denote a non-error, non-null result.

$$\text{ESR} = \frac{1}{N} \sum_{i=1}^{N} \prod_{j=1}^{k_i} \mathbf{1}[\text{OK}(r_{i,j})]. \tag{2}$$

Full Orchestration Success Rate (OSR). A sample is successful if all skills succeed:

$$\text{OSR} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[\text{TSR}_i = \text{OA}_i = \text{PPS}_i = \text{EGA}_i = 1] \,.$$

C.4 Tool Selection

Let

$$\mathcal{T}_i^* = \{t_{i,1}^*, \ldots, t_{i,m_i}^*\}, \quad \hat{\mathcal{T}}_i = \{\hat{t}_{i,1}, \ldots, \hat{t}_{i,k_i}\}.$$

Tool Selection Precision (TSP).

$$\text{TSP} = \frac{1}{N} \sum_{i=1}^{N} \frac{|\mathcal{T}_i^* \cap \hat{\mathcal{T}}_i|}{|\hat{\mathcal{T}}_i| + \varepsilon}. \tag{3}$$

Tool Selection Recall (TSR).

$$\text{TSR} = \frac{1}{N} \sum_{i=1}^{N} \frac{|\mathcal{T}_i^* \cap \hat{\mathcal{T}}_i|}{|\mathcal{T}_i^*|}. \tag{4}$$

C.5 Ordering and Workflow Planning

Ordering Accuracy (OA). Let $\text{Subseq}(A, B) = 1$ iff A is a subsequence of B.

$$\text{OA} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[\text{Subseq}(T_i^*, \hat{T}_i)\right]. \quad (5)$$

This measures whether the model preserves the logical order of tools.

C.6 Parameter Planning

Define validity of a tool call result:

$$\text{Good}(r) = \mathbf{1}[r \neq \text{NULL},\ r \neq \text{ERROR}].$$

Parameter Planning Success (per instance). For each gold-required tool class c, consider the model's first call to a tool in c and its result r.

$$\text{PPS}_i = \prod_{c \in \mathcal{C}_i^*} \text{Good}(r_{i,c}), \quad (6)$$

where $\mathcal{C}_i^*$ is the set of required tool classes.

Dataset-level PPS.

$$\text{PPS} = \frac{1}{N} \sum_{i=1}^{N} \text{PPS}_i.$$

Parameter Planning Tool Success (per tool-class).

Let M be total number of required tool classes evaluated:

$$\text{PPTS} = \frac{1}{M} \sum_{(i,c)} \text{Good}(r_{i,c}).$$

C.7 Entity Grounding

Let E_i^* be entity set from question + gold program, and $\hat{E}_i$ the entities implied by tool calls.

$$\text{EGA} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[\exists e \in \hat{E}_i \text{ matching some } e^* \in E_i^*\right].$$

C.8 Depth-Aware Planning

Mean Absolute Error.

$$\text{Depth-MAE} = \frac{1}{N} \sum_{i=1}^{N} |\hat{d}_i - d_i^*|.$$

Depth Bias.

$$\text{DB} = \frac{1}{N} \sum_{i=1}^{N} (\hat{d}_i - d_i^*).$$

C.9 Attempts and Recovery

Attempts-to-Success (ATS).

Let s_i be the index of the first successful call.

$$\text{ATS} = \frac{1}{N} \sum_{i=1}^{N} s_i.$$

C.10 Tool Explosion Diagnostics

Let K be tool-set size.

- Acc–T curve: $\text{Acc}(K)$
- Tool Selection Error Rate: $\text{TSER} = 1 - \text{TSP}$
- Depth-conditioned accuracy: $\text{Acc}(d^* \geq L)$

C.11 Case Study

C.11.1 Case Study I: Do functionally redundant tools unlock alternative execution paths?

Claude-Sonnet-4.5 achieves higher accuracy under OraA-3 tool exposure than under the oracle Oracle setting. Since OraA-3 differs from Oracle primarily by adding functionally redundant tools, we ask:

> **Do redundant tools unlock alternative executable workflows that are unavailable under the oracle minimal tool set?**

Experimental probe. We evaluate GPT-5.1 in a controlled setting where the agent is given the oracle tool set $\mathcal{G}$ augmented with redundant tools. We then search for instances satisfying:

1. the oracle tool set alone is sufficient to solve the task; and
2. the model nevertheless solves the task through a different workflow that explicitly uses redundant tools.

Across 1,524 questions we identify six such cases. Although rare, they are conceptually important: current agents seldom explore alternative workflows, yet multi-path execution does occur and can change measured accuracy under Top-k exposure.

Representative examples. Figure 10 illustrates two representative cases.

(a) Alternative executable paths. The agent sometimes constructs a workflow different from the golden program but semantically equivalent, showing that the benchmark implicitly admits multiple

valid execution paths even when only one is annotated.

(b) Redundant tools as rescue routes. In other cases, the canonical workflow fails due to missing data or transient API errors, while the agent succeeds by switching to a redundant tool. Redundancy thus acts as a fallback mechanism and improves robustness, rather than serving only as distraction.

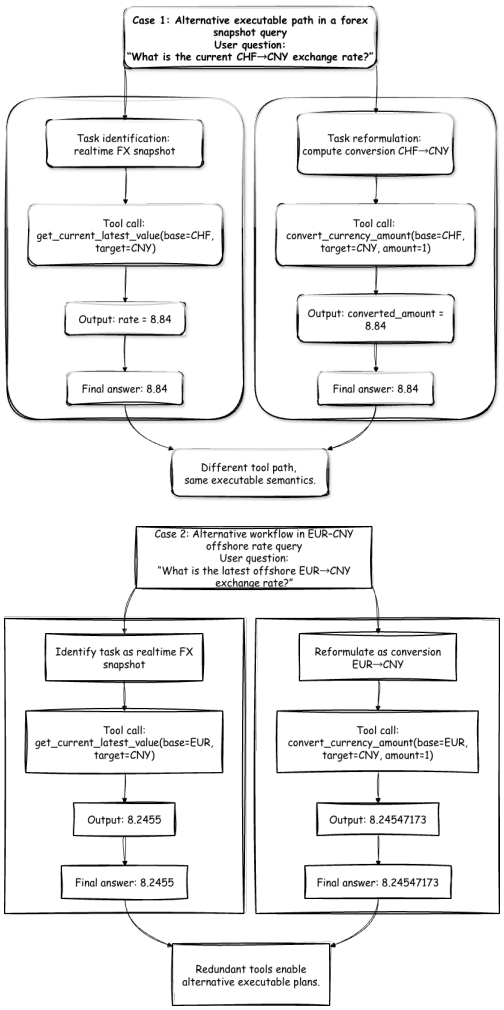

Figure 10: **Two examples of multi-path execution unlocked by redundant tools.** Top: an FX snapshot query solved through an alternative API chain; Bottom: a dividend query where redundant tools provide a fallback path after canonical-tool failure.

Takeaway. This case study suggests that:

- redundant tools occasionally unlock alternative valid execution paths that are not used under Oracle settings;
- current agents do not systematically search workflow space, explaining the small number of observed cases;
- when such paths are discovered, they can improve accuracy under Top-K tool exposure.

Taken together with Case Study II, this reveals a nuanced picture: moderate Top-K exposure may help by increasing the number of valid paths, whereas excessive tool exposure may hurt by increasing decision uncertainty and abstention. The key challenge is therefore not only to provide tools, but to guide agents in exploring the space of executable workflows effectively.

C.11.2 Case Study II: Does a Larger Top-K Make Agents Stronger or More Hesitant?

In the previous case study, we showed that OraA-3 exposure can outperform the oracle Oracle setting because it provides additional valid execution paths: agents sometimes reach the correct answer through workflows that differ from the reference program. We now ask a complementary question: **if increasing K creates even more potential paths, why does OraA-10 often perform worse than OraA-3?**

Experimental setup. We focus on the 394-instance paired subset. Each instance is evaluated twice with identical prompts, evaluation protocol, and models; the only difference is the size of the tool list presented to the agent ($k = 3$ vs. $k = 10$). This paired design isolates the effect of Top-K exposure from all other confounders.

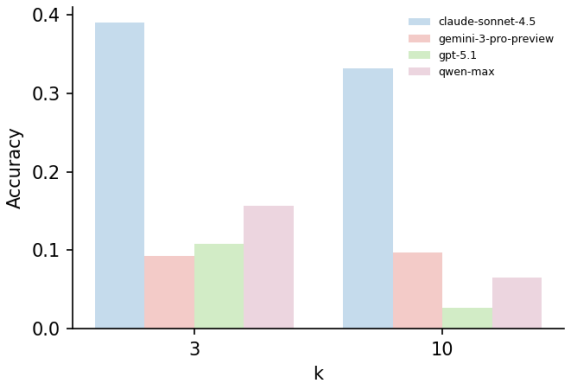

Figure 11: Accuracy comparison between $k = 3$ and $k = 10$ on the 394-item paired subset. Each bar shows mean accuracy across samples for each model.

Observation 1: Increasing K reduces accuracy. Figure 11 shows that OraA-3 consistently outperforms OraA-10 across models on the same tasks. This result is striking because OraA-10 strictly contains OraA-3: if tool availability were the primary

bottleneck, accuracy should be non-decreasing in K. Instead, expanding the visible tool set degrades performance, suggesting that the dominant factor is not the absence of the right tool, but the agent's behavior under increased choice.

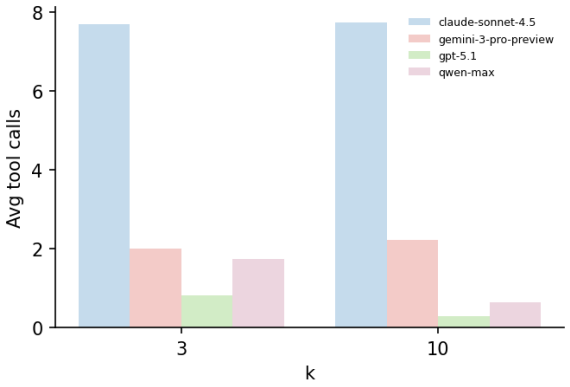

Figure 12: Change in average number of tool calls when increasing from $k = 3$ to $k = 10$. Positive values indicate more tool usage under larger k.

Observation 2: Lower accuracy is not caused by insufficient tool use. Figure 12 shows that the mean number of tool calls does not decrease when moving from $k = 3$ to $k = 10$; in many models it even increases. Therefore, OraA-10 does not fail simply because agents "give up early" or "call too few tools". Instead, errors arise during execution—mis-parameterization, tool mis-selection among near-duplicates, or inconsistent multi-step composition—rather than from a lack of action.

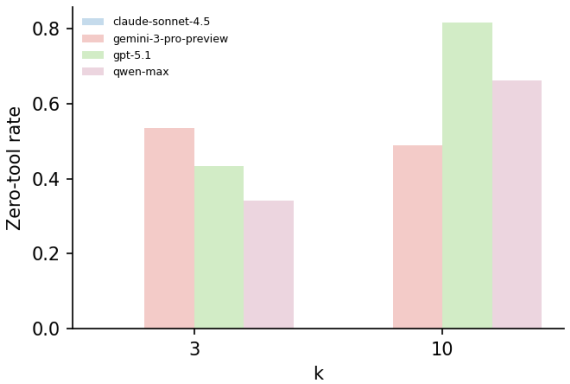

Figure 13: Zero-tool answer rate for $k = 3$ and $k = 10$. Higher values indicate a stronger tendency to avoid tool usage entirely.

Observation 3: Larger K increases hesitation and abstention. Figure 13 shows that OraA-10 settings exhibit a markedly higher zero-tool answer rate than OraA-3. Thus, enlarging the tool set does not only change which tools are used; it changes whether the agent decides to use tools at all. This indicates a shift in strategy: when faced with a larger and more redundant tool space, agents increasingly default to direct answers instead of committing to multi-step tool execution.

Interpretation. Putting these observations together, we obtain the following picture:

- Oracle $\rightarrow$ OraA-3 improves performance by introducing additional valid execution paths.
- OraA-3 $\rightarrow$ OraA-10 degrades performance by inflating decision uncertainty and abstention pressure.

In other words, tool exposure has a non-monotonic effect: a moderate number of tools increases the chance of discovering a successful workflow, but excessive tool exposure induces hesitation, increases zero-tool responses, and amplifies composition errors in deep tool chains.

Takeaway. The limiting factor for current agents is therefore not simply tool availability, but their robustness under choice overload. More tools can make agents not only more error-prone, but also more hesitant to act—highlighting tool explosion as a behavioral phenomenon rather than a purely algorithmic scaling issue.

C.11.3 Case Study III: Does coarse tool packaging eliminate the tool-explosion problem?

Top-K exposure limits the visible tool set but does not restructure it. A complementary strategy is to reorganize tools hierarchically into coarse functional packages and require the agent to first select an intent, then a package, and only then a specific tool. This raises the guiding question of this case study:

> **If tools are hierarchically packaged, does tool explosion essentially disappear?**

Hierarchical routing and packaging in practice. Figure 14 shows a representative execution trace under our three-stage routing framework: (i) coarse-grained intent classification, (ii) fine-grained intent routing, and (iii) tool selection within a constrained package. As illustrated in the figure, even after packaging, the agent must still choose among multiple highly similar tools inside the same package and reason over heterogeneous argument

schemas and return formats. Thus, packaging restructures the search space, but does not collapse it into a single deterministic route.

Aggregate empirical effect. We evaluate GPT-5.1 under this packaged regime and obtain 180 correct answers out of 520 questions (approximately 30% accuracy). This confirms that hierarchical packaging does help: compared with flat tool exposure, the agent faces fewer irrelevant tools and the routing decisions visualized in Figure 14 become more organized and interpretable.

But tool explosion persists. However, the same execution trace in Figure 14 also reveals why accuracy remains far from satisfactory. Most errors arise from:

- confusion among multiple tools within the same functional family,
- unstable parameter grounding across stages of the workflow,
- failure to normalize heterogeneous output schemas across tools,
- early abandonment of tool use when latency or API noise is encountered.

The branching structure shown in Figure 14 indicates that even after packaging, the agent still faces a large workflow search space rather than a single correct pipeline.

Takeaway. Coarse tool packaging mitigates surface-level search overload but does not eliminate tool explosion. The challenge is structural: the agent must coordinate multi-step workflows across noisy, partially redundant tools. Therefore, tool explosion is not merely a catalog-design artifact; it persists even under hierarchical routing, as visually evidenced in Figure 14.

C.11.4 Case Study IV: Why does accuracy collapse completely beyond execution depth 1?

Across all three subsets (128 / 249 / 394) and all exposure conditions (Oracle, Top-3, Top-10, full), we consistently observe the same phenomenon: tasks with execution depth 1 maintain non-zero accuracy, whereas accuracy rapidly declines as depth increases, and collapses to zero at depth 5. This is not an artifact of specific models or tool exposure settings; rather, it suggests a structural failure mode of current LLM-based agents.

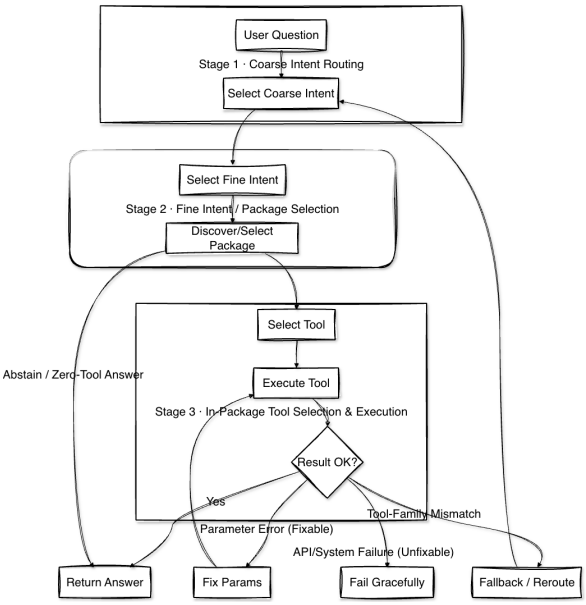

Figure 14: **Hierarchical intent routing and packaged tool selection.** The execution trace demonstrates that even when tools are grouped into packages, the agent must still disambiguate between multiple similar tools and coordinate multi-step workflows, indicating that tool explosion persists in packaged settings.

Figure 15: Zero-tool answer rate for $k = 3$ and $k = 10$. Higher values indicate a stronger tendency to avoid tool usage entirely.

Figure 15 illustrates this monotonic depth-induced degradation: depth 1 tasks are solvable at modest rates, but depth 5 tasks are solved by none of the evaluated agents under any tool-exposure configuration.

Observation 1: Failure is depth-driven, not exposure-driven. Notably, the collapse at depth 5 occurs even in the Oracle (oracle tool set) condition where tool selection burden is eliminated. The same pattern appears under Top-3, Top-10, and full exposure. This indicates that the source of difficulty is not merely tool retrieval or selection; instead, execution depth itself is the dominant factor.

Observation 2: Depth increases tool-use behavior but not accuracy. Across settings, the mean number of tool calls increases with execution depth, yet accuracy simultaneously decreases. This dissociation shows that additional tool use does not

translate into improved correctness: agents attempt longer workflows, but cannot maintain coherent execution over multiple steps.

Observation 3: Divergent failure behaviors at high depth. At depth 5, we observe two distinct behavioral responses depending on tool exposure: (i) under full (full-tool exposure), the zero-tool rate rises to around 50%, suggesting deliberate abstention in the face of overwhelming search space; (ii) under Top-K exposure, zero-tool rates drop to nearly zero while tool calls surge, indicating thrashing behavior where the agent repeatedly invokes tools without converging on the correct workflow. These contrastive behaviors point to a common bottleneck—long-horizon planning—modulated by exposure-induced uncertainty.

Why does depth hurt so strongly? The empirical evidence supports a multi-step consistency explanation:

- **state drift**: intermediate tool outputs must be remembered and reused, but partial results are overwritten or inconsistently referenced across steps;
- **temporal misalignment**: trading-day offsets and date normalization must remain consistent across calls, and small inconsistencies compound over depth;
- **parameter propagation errors**: arguments derived from previous tool outputs are misparsed, reformulated, or silently dropped in later calls;
- **error accumulation**: small inaccuracies at early steps propagate forward and dominate the final answer;
- **goal myopia**: agents locally repair tool errors without maintaining a global workflow plan, leading to the high-call "thrashing" observed at depth 5.

Takeaway. Execution depth is the single strongest predictor of failure in REALFIN. Even when tool selection is simplified (Oracle) or tool space is constrained (Top-K), agents fail beyond one or two steps. This suggests that the key challenge in financial agent evaluation is not isolated tool use, but maintaining globally consistent, temporally aligned, multi-step workflows over long horizons. Long-chain orchestration—rather than single-step competence—is therefore the dominant bottleneck for current LLM agents.

C.11.5 Case Study V: Representative failure types

Type	Representative example
Tool selection failure	*Question:* latest Shanghai silver benchmark price at noon. *Gold:* 17140.0. Oracle: calls `get_sge_benchmark_price` and answers correctly. 10-D / full: calls no tool and outputs free-form numbers.
Temporal grounding failure	*Question:* highest price of Suolian Co. 7 trading days ago. *Gold:* 33.56. Correct tool called, but indexes wrong trading day (returns 33.00).
Execution / answer grounding failure	*Question:* latest closing price of Yilite. *Gold:* 14.06. Correct tools called, but model ignores the "close" field and outputs unrelated values.

Table 12: **Representative failure types.** Tool explosion shifts error structure from computation to (i) tool selection, (ii) temporal grounding, (iii) answer grounding.

We illustrate how tool explosion changes the error landscape. Failures are not random: they cluster into a small number of systematic types that align with our process metrics.

D Alignment with Real-World Financial SOPs

We present three representative cases to illustrate that RealFin golden workflows closely mirror real-world financial standard operating procedures (SOPs). The cases cover (1) equity quarterly return computation, (2) cross-sectional liquidity aggregation, and (3) precious-metals benchmark price comparison.

E Supplement Experiment Results

Table 16: Model-averaged bootstrap results

Setting	ACC	Exec	Oracle	Prec	Rec
Oracle	0.21 ± 0.03	0.67 ± 0.03	0.76 ± 0.03	1.00 ± 0.00	0.68 ± 0.04
OraA-3	0.09 ± 0.02	0.34 ± 0.02	0.39 ± 0.02	0.98 ± 0.01	0.55 ± 0.04
OraA-10	0.05 ± 0.01	0.24 ± 0.02	0.25 ± 0.02	0.91 ± 0.02	0.35 ± 0.04
Pkg	0.28 ± 0.04	0.34 ± 0.01	0.00 ± 0.00	0.12 ± 0.03	0.13 ± 0.03
BM25*	0.13 ± 0.03	0.60 ± 0.04	0.64 ± 0.04	0.99 ± 0.01	0.43 ± 0.04
Full	0.12 ± 0.02	0.49 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00

Table 13: Equity quarterly return SOP vs. RealFin workflow (Tencent Holdings, Q4 2023).

Analyst SOP	RealFin golden workflow
Identify target security	`get_code("Tencent")`.
Determine quarter start and end dates	Use 2023-09-30 and 2023-12-31.
Retrieve adjusted closing prices	`get_price(close)`.
Ensure price comparability	`adjust = qfq`.
Compute period return	$\frac{P_{\text{end}}}{P_{\text{start}}} - 1$.
Format output	Percentage form, two decimals.

Table 14: Cross-sectional liquidity SOP vs. RealFin workflow (U.S. semiconductor "big three").

Analyst SOP	RealFin golden workflow
Define peer group	`search_kg("U.S. semiconductor big three")`.
Obtain constituent tickers	`get_code_from_name` for each member.
Select reference trading day	`query_date = latest`.
Retrieve trading volume	`get_price(volume)`.
Construct cross-sectional sample	Collect volumes across constituents.
Compute group-level statistic	Arithmetic mean of volumes.

Table 15: Precious-metals fixing SOP vs. RealFin workflow (Shanghai gold benchmark, 2024-05-15).

Analyst SOP	RealFin golden workflow
Specify asset and benchmark	`get_price(gold)`.
Choose pricing sessions	`session = morning / evening`.
Fix calendar date	`query_date = 2024-05-15`.
Retrieve benchmark prices	Two calls to `get_sge_benchmark_price`.
Compute intraday difference	$P_{\text{morning}} - P_{\text{evening}}$.
Reporting format	Return price difference.

Metric	Rate	Meaning
Retrieval fail	0.72	gold tools not retrieved
Orchestration fail	0.03	wrong order / composition
Param/temporal fail	0.17	wrong date or arguments
Full* success	0.08	correct final answer
TSR	0.15	all gold tools selected
OSR	0.03	Full* workflow correct

Table 17: All values are proportions over evaluated samples. OSR $\ll$ TSR indicates difficulty lies in multi-tool orchestration rather than single-tool use.

E.1 Error analysis

E.1.1 Quantitative error breakdown

Figure 16 shows that the majority of errors are caused by *tool selection* (78.1%), with execution errors (10.5%) and residual errors (9.0%) forming most of the remainder. Parameter, temporal, and aggregation errors jointly account for less than 3%. Thus the primary bottleneck in REALFIN is deciding which tools to use, not arithmetic or aggregation.

This pattern is consistent across models (70–84% selection error) and exposure regimes: Full and OraA-10 exhibit the highest selection error rates, whereas OraA-3 and BM25 regimes contain more execution mistakes.

E.1.2 Conditional performance given correct skills

We compute accuracy (i) conditioned on correct tool selection and (ii) conditioned on both correct selection and reference-consistent orchestration (Figure 17).

Main observation. Across all models, $\text{ACC} \mid \text{Selection}$ is 3–6$\times$ higher than raw accuracy, whereas $\text{ACC} \mid \text{Orchestration} \approx \text{ACC} \mid \text{Selection}$. Hence, once the right tools are chosen, ordering and arithmetic rarely fail; the dominant source of failure is tool selection under tool explosion.

F Annotation Guidelines

F.1 Human evaluation procedure

This appendix describes the human evaluation used for RealFin. The goal of the questionnaire is to assess the quality of tasks and reference workflows in the dataset. Annotators are instructed to read the guidelines carefully and base their judgments only on the information provided for each item.

For every item, annotators are given:

- the question description
- the golden answer
- the reference program implementing the reference workflow
- tool schema documentation, including tool names, functionality, and input and output fields

Annotators are asked not to rely on any external resources and not to execute the code. All judgments must be made purely from the provided information.

Model	Context	ACC	TSR	OSR	PPS
Claude	Short	0.300	0.500	0.067	0.467
Claude	Long	0.033	0.167	0.000	0.033
Qwen	Short	0.033	0.333	0.100	0.300
Qwen	Long	0.000	0.133	0.000	0.067

Table 18: Short vs. long context (fixed tool set; 30 tasks each).

Model	k	ACC	TSR	OSR
Qwen	0	0.1016	0.1040	0.0000
Qwen	3	0.1406	0.4844	0.1094
Qwen	6	0.0781	0.0938	0.0000
Qwen	10	0.0469	0.0859	0.0000

Table 19: Intervention on tool-space size (k distractors), fixed prompt context.

F.2 Rating dimensions

Each item is scored along four dimensions:

1. **Question reasonableness:** clarity, lack of ambiguity, and plausibility within real financial analysis scenarios
2. **Answer correctness:** consistency of the golden answer with the question and whether the value or conclusion is correct or essentially correct
3. **Reference workflow correctness:** without running the code, whether the tools are appropriate, parameters are reasonable, and the sequence of steps is conceptually able to produce the correct answer
4. **Tool schema usability:** clarity and sufficiency of the tool documentation, including input and output meanings, to reasonably support solving the task

F.3 Scoring scheme

All four dimensions use a three-level scale:

2 fully correct or reasonable 1 minor issues but usable 0 clearly incorrect or unusable

Annotators may optionally add short comments explaining their scores. If a judgment truly cannot be made due to insufficient information, annotators may select unable to judge and briefly explain the reason. Annotators complete the questionnaire independently.

F.4 Example annotation item

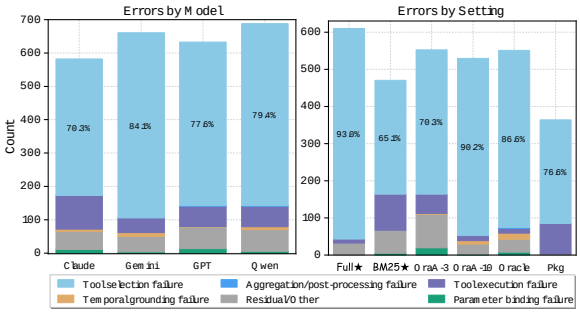

Figure 16: Error distribution.

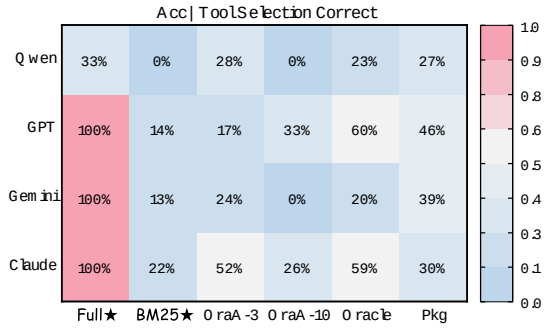

Figure 17: **Conditional accuracy.** Accuracy improves sharply once correct tools are selected.

Table 20: Example human evaluation item

ID	152
Question	In 2023 the year-on-year growth rate of the M2 money supply was 12.6% in January and changed to another value in December. Compute the change between January and December to two decimal places and report it as a percentage.
Golden answer	-1.8%
Reference program	abbreviated for space
Task type Time sensitivity	T2_Compare Fixed
Example ratings	Question 2 Answer 2 Workflow 2 Tool schema 2