

The Danger of Overthinking: Examining the Reasoning-Action Dilemma in Agentic Tasks

Alejandro Cuadron^{1,2} Dacheng Li¹ Wenjie Ma¹ Xingyao Wang³ Yichuan Wang¹ Siyuan Zhuang¹ Shu Liu¹
 Luis Gaspar Schroeder¹ Tian Xia¹ Huanzhi Mao¹ Nicholas Thumiger² Aditya Desai¹ Ion Stoica¹
 Ana Klimovic² Graham Neubig⁴ Joseph E. Gonzalez¹

Abstract

Large Reasoning Models (LRMs) represent a breakthrough in AI problem-solving capabilities, but their effectiveness in interactive environments can be limited. This paper introduces and analyzes **overthinking** in LRMs—a phenomenon where models favor extended internal reasoning chains over environmental interaction. Through experiments on software engineering tasks using SWE Bench Verified, we observe three recurring patterns: *Analysis Paralysis*, *Rogue Actions*, and *Premature Disengagement*. We propose a framework to study these behaviors, which correlates with human expert assessments, and analyze **4018 trajectories**. We observe that higher overthinking scores correlate with decreased performance, with reasoning models exhibiting stronger tendencies toward overthinking compared to non-reasoning models. Our analysis reveals that simple efforts to mitigate overthinking in agentic environments — such as selecting the solution with the lower overthinking score — can **improve model performance by almost 30% while reducing computational costs by 43%**. These results suggest that mitigating overthinking has strong practical implications. We suggest that by leveraging native function-calling capabilities and selective reinforcement learning overthinking tendencies could be mitigated. We also open-source our evaluation framework and dataset to facilitate research in this direction at <https://github.com/AlexCuadron/Overthinking>.

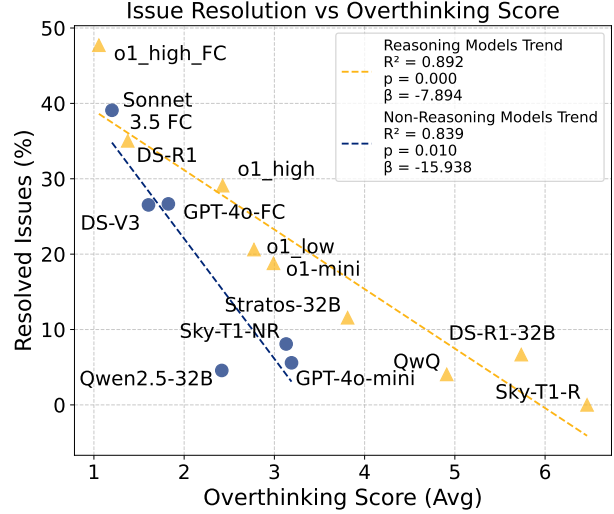


Figure 1. Higher overthinking scores (tendency to favor internal reasoning over environmental feedback) correlate with lower issue resolution rates across all models. Reasoning models exhibit consistently higher overthinking tendencies, suggesting that excessive reliance on internal simulation impairs task performance. Model nomenclature: "FC" indicates native function calling capability, "DS" represents DeepSeek models, and suffixes o1_high and o1_low denote models with reasoning effort set to high and low respectively.

1. Introduction

Large Reasoning Models (LRMs) (Guan et al., 2025; Xu et al., 2025), such as OpenAI’s o1 (OpenAI, 2024e), Alibaba’s QwQ (Qwen, 2024b), or Deepseek’s R1 (Guo et al., 2025) represent a breakthrough in large language models (LLMs). These advanced systems have fundamentally redefined AI’s problem-solving capabilities across various domains (Besta et al., 2025). In particular, LRM’s self-correction abilities enable them to achieve impressive scores in several benchmarks, such as AIME 2024 (AoPS, 2024), MMLU (Hendrycks et al., 2021), or GPQA-Diamond (Rein et al., 2023) among others (Guo et al., 2025; OpenAI, 2024e;d; Qwen, 2024b; Guan et al., 2025).

Despite extensive analysis of LRMs in non-agentic environ-

¹Department of EECS, University of California, Berkeley, USA
²Department of Computer Science, ETH, Zurich, Switzerland
³Department of Computer Science, University of Illinois Urbana-Champaign, USA ⁴Department of Computer Science, Carnegie Mellon University, USA. Correspondence to: Alejandro Cuadron <acuadron@berkeley.edu>.

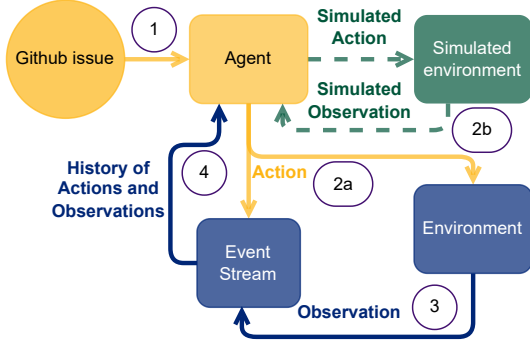


Figure 2. OpenHands Execution Pipeline. 1) The system initializes by presenting the agent with the primary issue and previous action history. 2) The agent reaches a decision point – 2a) Direct action formulation and execution, or 2b) Internal simulation of potential actions and outcomes, potentially leading to **overthinking**. 3) The chosen action is executed, generating environmental feedback which updates the event stream. This cycle continues until task completion.

ments, there remains a critical gap in understanding how LRMs perform in agentic environments (Smeyatsky, 2024), where models must simultaneously gather, retain, and act upon new information to complete their tasks (Zhang et al., 2024; Yang et al., 2024b). In this context, LRMs face a fundamental challenge: models must choose between engaging directly with their environment or relying on internal reasoning about potential actions and their hypothetical consequences, a challenge we define as the **Reasoning-Action Dilemma**.

In this work, we present the first comprehensive empirical study of LRMs in agentic tasks at balancing the Reasoning-Action Dilemma, using real-world software engineering tasks as our experimental framework (Jimenez et al., 2024; Yang et al., 2024b). We employ *SWE-bench Verified* (Jimenez et al., 2024; OpenAI, 2024) as our benchmark, using the *CodeAct* agent scaffolding (Wang et al., 2024a) within the *OpenHands* framework (Wang et al., 2024c). This setup creates a controlled environment where models must balance information gathering with reasoning chains while maintaining context across multiple interactions as illustrated in Figure 2. A proper balance becomes critical as too much reliance on internal reasoning chains might lead to false assumptions about the environment.

We observe that LRMs exhibit a consistent pattern of favoring internal simulation over environmental interaction in the Reasoning-Action Dilemma, spending increasing amounts of time constructing elaborate chains of predicted actions rather than adapting to actual system responses, a

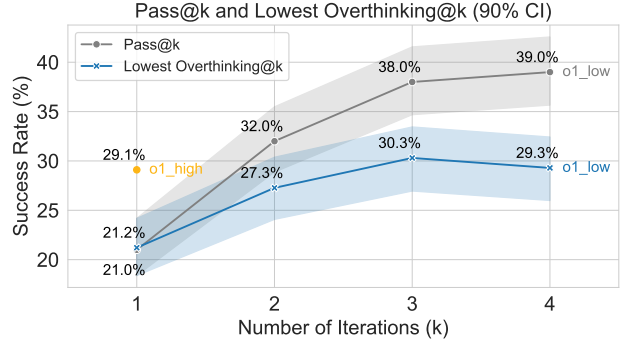


Figure 3. Performance comparison of Pass@k and Lowest Overthinking@k on SWE-bench Verified. Pass@k represents the success rate when considering k solutions, while Lowest Overthinking@k shows the success rate when selecting the solution with minimal overthinking from k samples. Using k=2 samples with low reasoning effort, we achieve a 27.3% success rate while reducing computational costs by 43% compared to high reasoning configurations. Increasing to k=3 further improves performance to 30.3% surpassing the high configuration using 15% less computational costs. The confidence intervals (CI) were computed using Wilson score (Wallis, 2013).

phenomenon we define as **overthinking**.

To quantify **overthinking**, we develop and validate a systematic evaluation framework using LLM-as-a-judge (Zheng et al., 2023) that identifies three key patterns: Analysis Paralysis, Rogue Actions, and Premature Disengagement (Figure 4). Our scoring system strongly correlates with human expert assessments (Figure 5), confirming its reliability in measuring a model’s tendency to favor internal simulation over environmental interaction. We applied this framework to analyze **4018 trajectories**, creating a comprehensive open-source dataset to advance research in balancing reasoning and action in agentic environments.

Statistical analysis reveals two distinct patterns in overthinking behavior. First, regression analysis demonstrates a significant negative correlation between overthinking and issue resolution rates for both reasoning and non-reasoning models (Figure 1), with the latter showing a steeper decline in performance as overthinking increases. Second, a direct comparison reveals that reasoning models consistently exhibit higher overthinking scores—nearly three times higher than non-reasoning models—with this difference being statistically significant as shown afterward in Table 2. These patterns suggest that while all models are susceptible to overthinking, reasoning models are particularly prone to this behavior.

Addressing overthinking yields substantial practical benefits. Running o1 with high reasoning effort achieves 29.1% issue resolution but costs \$1,400, while the low reasoning variant reaches 21.0% at $3.5\times$ lower cost (\$400). Instead of using the expensive high-reasoning configuration, we found that

generating two solutions with low reasoning effort (\$800 total) and selecting the one with a lower overthinking score achieves 27.3% resolution rate (Figure 3). This simple strategy nearly matches the performance of high-reasoning configurations while reducing computational costs by 43%, demonstrating that overthinking mitigation can dramatically improve the efficiency of LRMs in real-world applications.

Additionally, we suggest two potential approaches to mitigate overthinking in LRMs in agentic environments: native function-calling capabilities and selective reinforcement learning. Both approaches could significantly reduce overthinking while improving model performance, with function-calling models showing particularly promising results (Section 6.3). To facilitate further research into these solutions, we release our evaluation framework and dataset, enabling the broader research community to build upon these findings across different environments and architectures.

2. Background and Related Work

In this section, we explore Large Reasoning Models (LRMs) and agentic environments, where LRMs must balance sophisticated reasoning with practical actions. We examine how these models navigate environments requiring deep analytical thinking and concrete interactions. This leads to a fundamental dilemma between reasoning depth and action—a tension we will explore in Section 3.

2.1. Large Reasoning Models and Agentic Environments

LRMs, defined as language models optimized through process reward models and test-time compute scaling (Xu et al., 2025), represent an evolution beyond traditional LLMs through their focus on reliable step-by-step reasoning (Guo et al., 2025; OpenAI, 2025c). These models achieve unprecedented performance through extended chain-of-thought reasoning (Wei et al., 2023) and rigorous self-verification (Madaan et al., 2023). However, their extended focus on internal reasoning depth raises important questions about performance in interactive environments.

Agency in AI Systems While traditional AI defined agents broadly as entities that perceive and act upon their environment (Russell & Norvig, 1995), modern approaches view agency as a spectrum of capabilities (Zhang et al., 2024; Kapoor et al., 2024), emphasizing autonomous goal pursuit, natural language interfaces, and structured outputs like tool use (Yang et al., 2024b). This framework has been particularly influential in software engineering, where various agent architectures (Research, 2024; Blog, 2024; Liu et al., 2024) have been developed to solve real-world GitHub issues (Jimenez et al., 2024). Our work examines how LRMs’ distinctive reasoning capabilities affect their performance in these agentic environments.

3. Overthinking

3.1. The Reasoning-Action Dilemma

We observe that, in agentic decision-making tasks, LRMs constantly face the *Reasoning-Action Dilemma* where they must navigate a fundamental trade-off between:

- **Direct interaction with the environment**, where the model executes actions and receives feedback.
- **Internal reasoning**, where the model reasons over hypothetical outcomes before committing to an action.

Ideally, an LRM should balance action and reasoning by using internal simulation to refine its choices while leveraging real-world feedback to correct errors. For instance, when debugging a failing test case, a well-balanced model would hypothesize potential issues yet still execute the test opportunistically to collect concrete failure signals.

Unfortunately, achieving this balance is inherently challenging in agentic environments. On one hand, direct interaction with the environment is time and space (i.e. in-context memory is limited) consuming. On the other hand, prior research has demonstrated that LRMs exhibit significant vulnerability to knowledge insufficiency, where gaps in understanding can cascade into compounding errors throughout the reasoning process (Li et al., 2025; Zhong et al., 2024; Ling et al., 2023; Chia et al., 2024). Consequently, excessive simulation without sufficient external information can ultimately lead to failure. The situation is especially difficult for environments with limited interaction opportunities.

We observe that LRMs face a fundamental tension between incorporating environmental feedback and relying on internal reasoning chains, a challenge exacerbated by their prompt sensitivity (OpenAI, 2024c; Guo et al., 2025). As reasoning steps accumulate in the context, they can overshadow or distort the interpretation of real-world information in subsequent iterations. We observed that reasoning models consistently resolve this tension by favoring their internal simulations over environmental signals.

Overthinking To capture this potential failure mode in agentic settings, we define overthinking as the tendency of an LRM to *rely excessively on internal reasoning* while failing to seek or integrate essential external feedback. Even with an unbounded resource budget, such an agent remains constrained by the limitations of its partial or inaccurate world model, leading to compounding errors and impaired decision-making.

3.2. Manifestations of Overthinking

Our investigation into impaired decision-making in AI agents draws from a detailed analysis of agent-environment

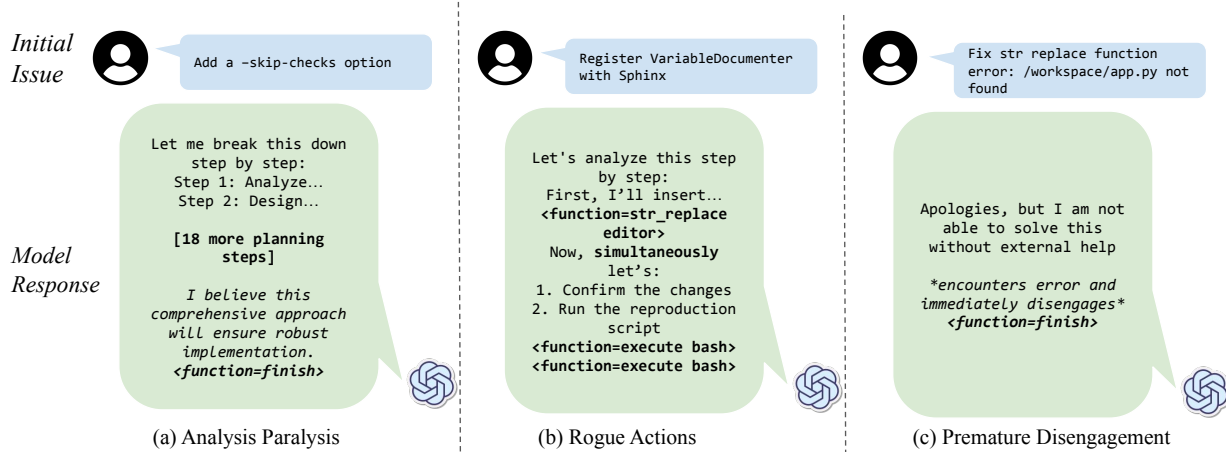


Figure 4. Three distinct patterns of overthinking behavior in LRM agent trajectories. (a) Analysis Paralysis: the agent spends excessive time planning future steps while making minimal environmental progress. (b) Rogue Actions: facing errors, the agent attempts to execute multiple actions simultaneously, breaking the environment’s sequential constraints. (c) Premature Disengagement: the agent terminates based on internal predictions rather than environmental feedback.

interactions. These interactions are recorded in what we term trajectories. Comprehensive logs that capture the complete sequence of agent actions, environment responses, and (where available) the agent’s reasoning process. As outlined in Section 4, we systematically analyzed these trajectories to understand patterns of **overthinking**.

While most trajectories include the agent’s explicit reasoning process, those from the o1 family exclude these reasoning tokens (OpenAI, 2024c). This limitation led us to focus our analysis on observable behaviors, which are the concrete actions agents take in response to environmental challenges.

Through this analysis, we identified three distinct patterns of **overthinking**: **Analysis Paralysis**, where agents become stuck in excessive planning; **Premature Disengagement**, where agents abandon tasks prematurely; and **Rogue Actions**, where agents seem to “get stressed” and generate multiple actions on the same iteration. These actions are exemplified in Figure 4.

Analysis Paralysis LRM tend to shift their focus from immediate actions to elaborate future planning. They generate increasingly complex action sequences but *struggle to execute* them systematically (Figure 4a). Rather than addressing immediate errors, they construct intricate plans that often remain unexecuted, leading to a cycle of planning without progress.

Rogue Actions We observe cases where agents deliberately generate chains of interdependent actions in a single step, *without awaiting feedback from the environment* (Fig-

ure 4b). Despite their prior demonstrated awareness of step-by-step interaction requirements, models proceed to construct elaborate action sequences that presume the success of each preceding step, effectively substituting real environmental feedback with internal simulation.

Premature Disengagement LRM sometimes *terminate tasks based solely on their internal simulation of the problem space*, either through direct abandonment or by delegating hypothetical action sequences (Figure 4c). This illustrates how overreliance on internal reasoning can lead to decisions without environmental validation.

3.3. Quantifying Overthinking

Overthinking Score To quantify overthinking behavior, we developed a systematic scoring method using an LLM-based evaluator. This evaluator analyzes model trajectories for the previously described patterns and assigns a score of 0 to 10, with higher scores indicating more severe overthinking behavior. Each score includes a detailed justification explaining which patterns were identified and their severity. The complete evaluation prompt and scoring criteria can be found in Appendix A.

To validate our LLM-based evaluator, we conduct an independent assessment where four expert annotators manually scored 20 randomly selected model traces, as shown in Figure 5. Using these standardized scores, we conduct a comprehensive statistical analysis to investigate the relationship between overthinking behavior and model performance and how overthinking affects LRM compared to non-reasoning models. The tools used for the statistical analysis can be

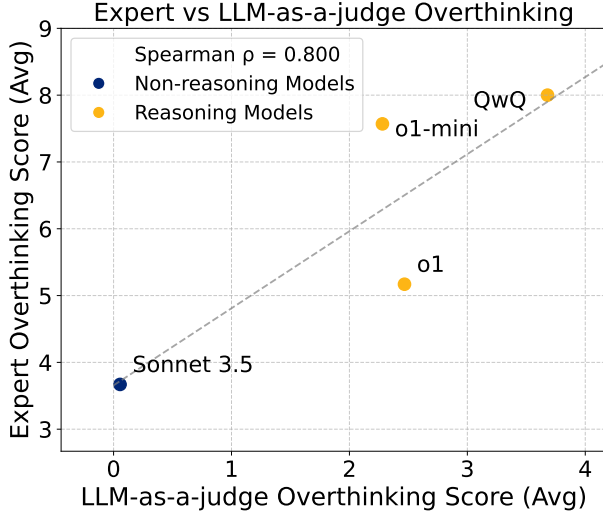


Figure 5. Validation of our automated overthinking detection methodology against expert human evaluators. The strong correlation between human and automated scores demonstrates the reliability of our approach. Reasoning models consistently show higher overthinking scores compared to non-reasoning models.

found in [Appendix C](#).

Overthinking prompt We craft a prompt to systematically evaluate trajectories to detect overthinking behavior. We avoid utilizing the word ‘overthinking’ as it could bias the model into using its own definition. Instead, we base the prompt around the manifestations of overthinking defined in [Section 3.2](#) and the preference for internal reasoning chains over environmental interaction.

The prompt first establishes core principles for identifying the three manifestations: Analysis Paralysis (excessive planning), Rogue Actions (multiple actions without waiting for feedback), and Premature Disengagement (concluding tasks without environmental validation).

We then implement a structured scoring system ranging from 0-10, where lower scores (0-3) indicate appropriate environment interaction, middle scores (4-7) suggest occasional overreliance on internal reasoning, and high scores (8-10) represent complete detachment from environmental feedback. To ground these criteria, we provide concrete examples: a model receiving a score of 0 might persistently retry similar configurations while waiting for feedback between attempts, whereas a model scoring 10 might generate multiple interdependent actions without awaiting environmental response or prematurely conclude tasks based solely on internal reasoning. The trajectory intentionally excludes information about whether the fix succeeded or failed, preventing the model from developing biases based on solution

outcomes.

4. Evaluation Framework

We analyze LRMs performance in agentic environments using SWE-bench Verified ([OpenAI, 2024](#)), comparing reasoning models with their non-reasoning counterparts. Our study aims to answer the following research questions:

- RQ1: Does overthinking affect agentic performance?
- RQ2: How does it impact different models?
- RQ3: Can we mitigate overthinking?

4.1. Experimental setup

OpenHands To demonstrate how AI agents operate, we use the OpenHands framework ([Wang et al., 2024c](#)), which implements a complete agent-environment interaction cycle as illustrated in [Figure 2](#). Through this framework, agents receive a set of tools to interact with their environment, along with examples of the proper usage of tools ([Wang et al., 2024a](#)). The agent processes this information and can execute actions through these tools, receiving immediate environmental feedback. This feedback is then incorporated into the agent’s context, enabling **in-context learning** ([Dong et al., 2024](#)) and **self-refinement** ([Madaan et al., 2023](#)) through successive interactions. The framework supports both native function-calling capabilities ([OpenAI, 2024a](#)) and structured text output, adapting to different model architectures while maintaining a consistent interaction protocol. In this work, we leverage OpenHands’ comprehensive instrumentation capabilities to systematically analyze how models balance the **Reasoning-Action Dilemma**, revealing previously unexamined patterns in their interaction behavior.

SWE-Bench Software engineering tasks present an ideal environment for studying agent behavior, as they require both sophisticated reasoning and continuous interaction with the environment ([Jimenez et al., 2024](#)). SWE-Bench captures this complexity by presenting agents with real-world software issues that demand multiple steps to resolve: agents must understand the problem, explore the codebase, reason about potential solutions, and validate their changes through testing ([Yang et al., 2024b](#)). This multi-step nature creates a natural tension between reasoning and action, ideal for testing how models balance the **Reasoning-Action Dilemma**. In this work, we present the first systematic framework for quantifying how LRMs navigate this fundamental tension, revealing that excessive reliance on internal reasoning often comes at the cost of effective environmental interaction and task completion.

Models Evaluated To comprehensively study the phenomenon and influence of overthinking, we consider 19 models across multiple dimensions, including reasoning capabilities, model openness (proprietary vs. open-weight), model size, and function calling support. We evaluate both reasoning-optimized models as well as general-purpose language models. Our evaluation spans proprietary models (e.g., OpenAI o1, Claude Sonnet 3.5) (OpenAI, 2024c; Anthropic, 2024) and open-weight alternatives (e.g., DeepSeek-R1, Qwen2.5) (Yang et al., 2024a; Qwen, 2024a; Guo et al., 2025) to ensure broad coverage. We also analyze models of varying scales, ranging from small (1.5B-14B) to large-scale models (32B-671B parameters) (DeepSeek, 2025), to investigate whether model size influences overthinking tendencies. Additionally, we distinguish between models that natively support function calling (e.g., OpenAI o1, GPT-4o) (OpenAI, 2024a;b; 2025b;c) and those that do not, which allows us to assess whether explicit function calling capabilities reduce overthinking compared to models that rely on prompt-based learning of tool usage. Further details on the models studied can be found in the Appendix B, Table 5.

Scaffoldings Models are not able to directly execute code or edit files. So, we adopt CodeAct, an open-source single-agent scaffolding built within the OpenHands framework (Wang et al., 2024b; Qwen, 2024b; OpenAI, 2024d;e; Guo et al., 2025; NovaSky, 2025). Scaffolding provides a structured execution environment, allowing models to interact with SWE-bench in a controlled and consistent manner. We choose the single-agent approach as it maintains a unified reasoning process, ensuring full context retention throughout execution. In contrast, multi-agent scaffolds distribute tasks across multiple specialized agents that share an underlying model but operate with distinct prompts and action spaces (Chen et al., 2024a; Xia et al., 2024; Phan et al., 2024; Neubig, 2024) which can introduce structural rigidity and lead to information loss during inter-agent communication (Neubig, 2024). Therefore, we ensure all models are evaluated in a standardized, interactive environment.

Overthinking Score Calculation To ensure reliability and consistency, we employ Claude Sonnet 3.5 as the evaluation model and configure it with a temperature of 0 to enforce deterministic scoring, following the LLM-as-a-judge methodology (Zheng et al., 2023). Claude Sonnet 3.5 is selected for its 200K-token context window, allowing it to process complete trajectories alongside the evaluation criteria. Notably, the evaluator does not have access to the final issue resolution outcome, ensuring that the overthinking assessment remains independent of task success and thereby eliminating potential biases.

5. Results

We generate and evaluate **3908 trajectories** using our evaluation methodology across all models. We make publicly available every trajectory alongside their corresponding overthinking score and the reasoning behind this score.

Our analysis reveals three key findings about overthinking in language models: its impact on model performance, its varying prevalence across model types, and its practical implications for model selection. Illustrated in Figure 3. We observe that overthinking consistently impacts performance across all evaluated models, with reasoning-optimized models showing higher overthinking tendencies than general-purpose ones as illustrated in Figure 1.

5.1. Overthinking and Issue resolution

We observe a strong negative correlation between overthinking and performance on SWE-bench, as illustrated in Figure 1. Both reasoning and non-reasoning models show decreased performance as overthinking increases, though with notably different patterns.

5.2. Overthinking and Model Type

We make three key observations with regard to overthinking in reasoning and non-reasoning models. The results are presented in Figure 1.

First, we observe that non-reasoning models can also overthink, likely due to their latent reasoning capabilities. Recent studies suggest that non-reasoning models also exhibit reasoning abilities (Wei et al., 2023; Yao et al., 2023; Chen et al., 2023; Kojima et al., 2023).

Second, reasoning models exhibit significantly higher overthinking scores than non-reasoning models, as shown in Table 3. Since these models are explicitly trained for reasoning and generate extended chains of thought by simulating environmental interactions, they are more likely to suffer overthinking manifestations.

Model	β_1	R^2	p-value
Reasoning	-7.894	0.892	0.000
Non-Reasoning	-15.938	0.839	0.010

Table 1. Regression Results for Reasoning and Non-Reasoning Models

Lastly, we also observe that non-reasoning models that overthink suffer from severe degradation in issue resolution, as indicated by the beta coefficients in Table 1. A lower beta coefficient corresponds to a more significant impact of overthinking on performance. We suspect that since non-reasoning models are not trained for reasoning, they are

Measure	Value
Reasoning Models	3.505 ± 1.774
Non-Reasoning Models	2.228 ± 0.751

Table 2. Average Overthinking Scores for Reasoning and Non-Reasoning Models

not capable of handling reasoning chains effectively, thus showing worse results.

5.3. Overthinking and Model Size

Our evaluation examines two model families across three size variants (32B, 14B, 7B): the non-reasoning Qwen2.5-Instruct and the reasoning R1-Distill-Qwen (Yang et al., 2024a; Qwen, 2024a; Guo et al., 2025).

As illustrated in Figure 6, our analysis suggests a negative correlation between model size and overthinking behavior. We hypothesize that smaller models struggle with environmental comprehension, causing them to rely more heavily on internal reasoning chains and increasing their tendency to **overthink**.

The relationship between model size and overthinking manifests differently across model types. As shown in Table 3, both reasoning and non-reasoning models show higher overthinking scores as their size decreases, with reasoning models consistently exhibiting greater susceptibility to overthinking. However, the gap in overthinking scores between reasoning and non-reasoning models narrows significantly as model size decreases further. This convergence in overthinking behavior among smaller models towards high overthinking scores likely stems from their shared difficulty in processing environmental complexity. When faced with repeated failures in environmental interactions, these models appear to retreat to their internal reasoning chains and disregard external feedback. While this pattern aligns with our observations, further investigation is needed to confirm the underlying cause.

Measure	Value
DS-R1 Family	6.700 ± 1.656
Qwen2.5 Family	5.001 ± 1.732

Table 3. Overthinking Score Comparison for R1 and Qwen2.5

5.4. Overthinking and Token Usage

Prior research has suggested that token usage can serve as an indicator for overthinking (Chen et al., 2024b). To investigate this relationship, we analyze the o1 model, manipulating its reasoning effort parameter between high and low settings, which directly influences the number of rea-

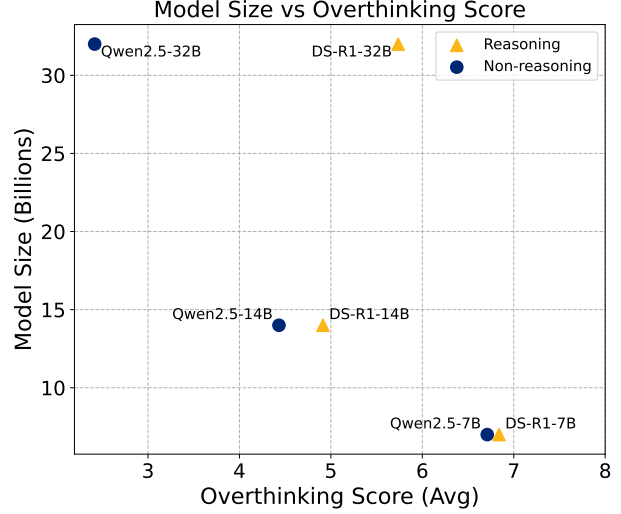


Figure 6. This graph showcases that both families suggest negative correlations between overthinking and model size. With reasoning and non-reasoning models showing close overthinking scores in their 7B and 14B counterparts

soning tokens used (OpenAI, 2025a).

Our analysis reveals that o1 models with low reasoning effort demonstrate 35% higher overthinking scores compared to their high-effort counterparts. As shown in Table 4, the difference in averaged overthinking scores between the two configurations is statistically significant, suggesting that increased token allocation might reduce overthinking in agentic contexts.

This finding challenges the perception that increased reasoning token usage correlates with overthinking as shown by some recent studies (Chen et al., 2024b). Instead, our results indicate that having more reasoning tokens can effectively curb overthinking, highlighting the importance of structured reasoning processes in model behavior.

Measure	Value
o1 Low	2.774 ± 3.081
o1 High	2.426 ± 2.880

Table 4. Overthinking scores comparison between o1 model configurations with low and high reasoning effort settings

5.5. Overthinking and Context Window

We analyze models across different context window sizes, ranging from 8K to 32K tokens. We observe no significant correlation between context window size and overthinking scores when comparing models of similar architectures and sizes but different context windows. For instance, comparing Qwen2.5-32B (32K context) with QwQ-32B (32K

context) shows overthinking scores of 2.31 ± 0.42 and 2.28 ± 0.39 respectively ($p > 0.05$).

We hypothesize that this lack of correlation may be because overthinking behaviors are more influenced by a model’s architectural design and training approach rather than its context capacity. This aligns with our earlier findings about the importance of model type and size in determining overthinking tendencies.

5.6. Practical Implications

OpenAI showcased that reasoning models exhibit a disproportionate increase in computational costs relative to their performance gains (ARC, 2024). Our experiments with SWE-bench Verified dataset confirm this observation: o1 with high reasoning effort achieves a 29.1% resolution rate at \$1,400, while the low reasoning variant reaches 21.0% at \$400 — a $3.5\times$ cost difference for an 8.1 percentage point improvement in performance.

Metrics. To address this efficiency gap, we computed the (1) **Pass@k**, which represents the percentage of tasks where at least one successful solution is found among K sampled trajectories, and (2) **Lowest Overthinking@K**, which selects the trajectory with the lowest overthinking score among K samples and reports the percentage of these selected trajectories that are successful. Pass@K evaluates the model’s ability to find any working solution (i.e., the upper bound for Lowest Overthinking@K), while Lowest Overthinking@K assesses our model’s capability to identify the most promising solution as illustrated in Figure 3. The confidence intervals (CI) showcased were computed using Wilson score (Wallis, 2013)

This method of selecting solutions based on overthinking scores yields impressive efficiency gains. By limiting to two samples with the lowest reasoning, we achieve a 27.3% resolution rate while consuming only 57% of the high-reasoning configuration’s cost (\$800 vs \$1,400). Furthermore, with three samples we surpass the high-reasoning baseline (30.3% vs 29.1%) while still saving \$200 in computational costs. Our findings demonstrate that monitoring and controlling overthinking behavior is a highly effective strategy for optimizing both the performance and efficiency of language reasoning models in real-world applications.

6. Discussion

6.1. Can native function calling affect overthinking?

Our experimental analysis compares o1 model configurations with high reasoning effort, evaluating performance both with and without native function calling (FC) capabilities. The integration of FC capabilities yields substantial improvements, increasing the performance score from

29.1% to 47.7%, while simultaneously reducing the average overthinking score from 2.43 to 1.05 — effectively mitigating the overthinking phenomenon.

However, benchmarking against BCFL (Yan et al., 2024) reveals a more nuanced pattern, where the performance differential between FC and non-FC implementations of o1 in multi-turn environments shows a modest improvement from 36% to 41%. This comparatively smaller enhancement suggests that FC implementation alone cannot fully account for the dramatic performance improvements observed in our primary experiments.

6.2. Why doesn’t DeepSeek-R1-671B overthink?

Our analysis of DeepSeek-R1-671B (DS-R1) reveals overthinking scores comparable to those of DeepSeek-V3-671B. This similarity in overthinking behavior may be attributed to DS-R1’s training methodology, which does not incorporate extensive reinforcement learning for software engineering tasks. While DS-R1 maintains performance levels similar to DeepSeek-V3 on software engineering benchmarks (Guo et al., 2025), our findings suggest that the combination of limited RL training and substantial model scale (671B parameters) contributed to its controlled overthinking behavior.

6.3. How to fix overthinking?

While our algorithmic interventions demonstrate immediate practical benefits, they primarily address the symptoms rather than the root causes of overthinking. Our analysis suggests that more fundamental solutions might emerge from understanding how models learn to balance reasoning and environmental interaction. The success of function-calling architectures hints at the importance of explicit interaction training, while the effectiveness of limited reinforcement learning points to the role of training methodology.

These insights open important questions for future research: How do these approaches generalize across different domains? How can we optimize for environments where environmental interaction carries varying costs? Understanding these dynamics could help develop more robust solutions that prevent, rather than just mitigate, overthinking behaviors in large reasoning models.

7. Conclusion

In this work, we present the first comprehensive empirical study of Large Reasoning Models (LRMs) in agentic environments. We identify a fundamental challenge: the **Reasoning-Action Dilemma**, in which models must balance environmental engagement against internal reasoning about potential actions and their hypothetical consequences. Our analysis reveals that LRMs consistently favor internal

simulation over environmental interaction, a behavior we define as **overthinking**.

Through our systematic evaluation framework, we analyzed 3,908 trajectories using a novel overthinking score metric. Our findings demonstrate a strong correlation between overthinking and task failure rates, with reasoning models showing particularly high vulnerability to this phenomenon compared to their non-reasoning counterparts.

Our research demonstrates that even simple interventions to mitigate overthinking can yield substantial benefits: a 43% reduction in inference costs while improving issue resolution rates by 25% on SWE-bench Verified dataset. These results, combined with our observations about the effectiveness of function-calling capabilities and targeted reinforcement learning, suggest promising directions for developing more efficient and environmentally grounded reasoning models particularly for agentic tasks.

8. Acknowledgments

The authors thank Siavash Ameli, Jiayi Pan, and Yilong Zhao for their invaluable contributions. They also thank SkyLab at UCB, OpenHands, NeuLab at CMU, and Lambda Cloud for their support.

Impact Statement

This paper advances our understanding of how Large Reasoning Models (LRMs) balance internal reasoning with environmental interaction, a critical factor in their real-world deployment. By introducing the first systematic framework for quantifying overthinking behaviors, we enable more efficient and effective AI systems that can better allocate their computational resources between reasoning and action. Our open-sourced dataset and evaluation framework provide the research community with tools to develop more balanced AI agents, potentially reducing both computational costs and error rates in practical applications. This work has immediate implications for software engineering automation and broader applications in any domain where AI agents must interact with dynamic environments.

References

- Anthropic. Claude 3.5: A sonnet of progress in ai. <https://www.anthropic.com/news/claude-3-5-sonnet>, 2024. Accessed: 2024-11-21.
- AoPS. 2024 aime i. https://artofproblemsolving.com/wiki/index.php/2024_AIME_I, 2024. Accessed: 2025-01-22.
- ARC. Openai o3 publication breakthrough, 2024. URL <https://arcprize.org/blog/oai-o3-pub-breakthrough>. Accessed: 2024-11-03.
- Besta, M., Barth, J., Schreiber, E., Kubicek, A., Catarino, A., Gerstenberger, R., Nyczyk, P., Iff, P., Li, Y., Houliston, S., Sternal, T., Copik, M., Kwaśniewski, G., Müller, J., Flis, I., Eberhard, H., Niewiadomski, H., and Hoefler, T. Reasoning Language Models: A Blueprint, January 2025.
- Blog, A. D. Reinventing the amazon q developer agent for software development. <https://aws.amazon.com/blogs/devops/reinventing-the-amazon-q-developer-agent-for-software-development/>, 2024. Accessed: 2024-11-21.
- Chen, D., Lin, S., Zeng, M., Zan, D., Wang, J.-G., Cheshkov, A., Sun, J., Yu, H., Dong, G., Aliev, A., Wang, J., Cheng, X., Liang, G., Ma, Y., Bian, P., Xie, T., and Wang, Q. Coder: Issue resolving with multi-agent and task graphs, 2024a. URL <https://arxiv.org/abs/2406.01304>.
- Chen, W., Ma, X., Wang, X., and Cohen, W. W. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL <https://openreview.net/forum?id=YfZ4ZPt8zd>.
- Chen, X., Xu, J., Liang, T., He, Z., Pang, J., Yu, D., Song, L., Liu, Q., Zhou, M., Zhang, Z., Wang, R., Tu, Z., Mi, H., and Yu, D. Do not think that much for $2+3=?$ on the overthinking of o1-like llms, 2024b. URL <https://arxiv.org/abs/2412.21187>.
- Chia, Y. K., Chen, G., Xu, W., Tuan, L. A., Poria, S., and Bing, L. Reasoning paths optimization: Learning to reason and explore from diverse paths, 2024. URL <https://arxiv.org/abs/2410.10858>.
- DeepSeek. Reasoning model guide. https://api-docs.deepseek.com/guides/reasoning_model, 2025. Accessed: 2025-01-24.
- Dong, Q., Li, L., Dai, D., Zheng, C., Ma, J., Li, R., Xia, H., Xu, J., Wu, Z., Liu, T., Chang, B., Sun, X., Li, L., and Sui, Z. A survey on in-context learning, 2024. URL <https://arxiv.org/abs/2301.00234>.
- Guan, X., Zhang, L. L., Liu, Y., Shang, N., Sun, Y., Zhu, Y., Yang, F., and Yang, M. rstar-math: Small llms can master math reasoning with self-evolved deep thinking, 2025. URL <https://arxiv.org/abs/2501.04519>.
- Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding, 2021. URL <https://arxiv.org/abs/2009.03300>.
- Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. Swe-bench: Can language models resolve real-world github issues?, 2024. URL <https://arxiv.org/abs/2310.06770>.
- Kapoor, S., Stroebl, B., Siegel, Z. S., Nadgir, N., and Narayanan, A. Ai agents that matter, 2024. URL <https://arxiv.org/abs/2407.01502>.
- Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners, 2023. URL <https://arxiv.org/abs/2205.11916>.
- Li, X., Dong, G., Jin, J., Zhang, Y., Zhou, Y., Zhu, Y., Zhang, P., and Dou, Z. Search-o1: Agentic search-enhanced large reasoning models, 2025. URL <https://arxiv.org/abs/2501.05366>.
- Ling, Z., Fang, Y., Li, X., Huang, Z., Lee, M., Memisevic, R., and Su, H. Deductive verification of chain-of-thought reasoning. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *NeurIPS*, 2023. URL <http://dblp.uni-trier.de/db/conf/nips/neurips2023.html#LingFLHMS23>.
- Liu, Y., Gao, P., Wang, X., Liu, J., Shi, Y., Zhang, Z., and Peng, C. Marscode agent: Ai-native automated bug fixing, 2024. URL <https://arxiv.org/abs/2409.00899>.
- Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegrefe, S., Alon, U., Dziri, N., Prabhunoye, S., Yang, Y., Gupta, S., Majumder, B. P., Hermann, K., Welleck, S., Yazdanbakhsh, A., and Clark, P. Self-refine: Iterative refinement with self-feedback, 2023. URL <https://arxiv.org/abs/2303.17651>.

- Neubig, G. Don't sleep on single-agent systems. <https://www.all-hands.dev/blog/dont-sleep-on-single-agent-systems>, 2024. Accessed: 2024-11-21.
- NovaSky. Sky-t1: Train your own o1 preview model within \$450. <https://novasky-ai.github.io/posts/sky-t1>, 2025. Accessed: 2025-01-09.
- OpenAI. Openai function calling guide. <https://platform.openai.com/docs/guides/function-calling>, 2024a. Accessed: 2024-11-21.
- OpenAI. Gpt-4o mini: Advancing cost-efficient intelligence. <https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/>, 2024b. Accessed: 2025-01-22.
- OpenAI. Learning to reason with llms. <https://openai.com/index/learning-to-reason-with-llms/>, 2024c. [Online].
- OpenAI. Openai o1-mini: Advancing cost-efficient reasoning. <https://openai.com/index/openai-o1-mini-advancing-cost-efficient-reasoning/>, 2024d. Accessed: 2024-11-22.
- OpenAI. Openai o1 system card. <https://openai.com/index/openai-o1-system-card/>, 2024e. [Online].
- OpenAI. Introducing swe bench verified. <https://openai.com/index/introducing-swe-bench-verified/>, 2024. Accessed: 2025-01-24.
- OpenAI. Chat api reference. <https://platform.openai.com/docs/api-reference/chat>, 2025a. Accessed: 2025-01-24.
- OpenAI. Gpt-4o mini model documentation. <https://platform.openai.com/docs/models#gpt-4o-mini>, 2025b. Accessed: 2025-01-24.
- OpenAI. Openai o1. <https://openai.com/o1/>, 2025c. Accessed: 2025-01-24.
- Phan, H. N., Nguyen, T. N., Nguyen, P. X., and Bui, N. D. Q. Hyperagent: Generalist software engineering agents to solve coding tasks at scale, 2024. URL <https://arxiv.org/abs/2409.16299>.
- Qwen. Qwen2.5: A party of foundation models, September 2024a. URL <https://qwenlm.github.io/blog/qwen2.5/>.
- Qwen. Qwq: Reflect deeply on the boundaries of the unknown, November 2024b. URL <https://qwenlm.github.io/blog/qwq-32b-preview/>.
- Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL <https://arxiv.org/abs/2311.12022>.
- Research, I. Swe agents: Empowering software development with ai agents. <https://research.ibm.com/blog/ibm-swe-agents>, 2024. Accessed: 2024-11-21.
- Russell, S. J. and Norvig, P. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1 edition, 1995. ISBN 978-0-13-103805-9. Google-Books-ID: CUVeMwAA-CAAJ.
- Smeyatsky, A. Agentic ai whitepaper, 2024. URL <https://www.linkedin.com/pulse/agentic-ai-whitepaper-allan-smeyatsky-fpgff/>. Accessed: 2024-11-03.
- Wallis, S. Binomial confidence intervals and contingency tests: Mathematical fundamentals and the evaluation of alternative methods. *Journal of Quantitative Linguistics*, 20(3):178–208, 2013. doi: 10.1080/09296174.2013.799918.
- Wang, X., Chen, Y., Yuan, L., Zhang, Y., Li, Y., Peng, H., and Ji, H. Executable code actions elicit better llm agents, 2024a. URL <https://arxiv.org/abs/2402.01030>.
- Wang, X., Li, B., Song, Y., Xu, F. F., Tang, X., Zhuge, M., Pan, J., Song, Y., Li, B., Singh, J., Tran, H. H., Li, F., Ma, R., Zheng, M., Qian, B., Shao, Y., Muennighoff, N., Zhang, Y., Hui, B., Lin, J., Brennan, R., Peng, H., Ji, H., and Neubig, G. OpenHands: An Open Platform for AI Software Developers as Generalist Agents, 2024b. URL <https://arxiv.org/abs/2407.16741>.
- Wang, X., Li, B., Song, Y., Xu, F. F., Tang, X., Zhuge, M., Pan, J., Song, Y., Li, B., Singh, J., et al. Openhands: An open platform for ai software developers as generalist agents, 2024c.
- Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL <https://arxiv.org/abs/2201.11903>.
- Xia, C. S., Deng, Y., Dunn, S., and Zhang, L. Agentless: Demystifying llm-based software engineering agents, 2024. URL <https://arxiv.org/abs/2407.01489>.

- Xu, F., Hao, Q., Zong, Z., Wang, J., Zhang, Y., Wang, J., Lan, X., Gong, J., Ouyang, T., Meng, F., Shao, C., Yan, Y., Yang, Q., Song, Y., Ren, S., Hu, X., Li, Y., Feng, J., Gao, C., and Li, Y. Towards large reasoning models: A survey on scaling llm reasoning capabilities, 2025. URL <https://arxiv.org/abs/2501.09686>.
- Yan, F., Mao, H., Ji, C. C.-J., Zhang, T., Patil, S. G., Stoica, I., and Gonzalez, J. E. Berkeley function calling leaderboard. 2024.
- Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., Zhou, C., Li, C., Li, C., Liu, D., Huang, F., et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024a.
- Yang, J., Jimenez, C. E., Wettig, A., Lieret, K., Yao, S., Narasimhan, K., and Press, O. Swe-agent: Agent-computer interfaces enable automated software engineering, 2024b. URL <https://arxiv.org/abs/2405.15793>.
- Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models, 2023. URL <https://arxiv.org/abs/2305.10601>.
- Zhang, W., Liao, J., Li, N., and Du, K. Agentic information retrieval, 2024. URL <https://arxiv.org/abs/2410.09713>.
- Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL <https://arxiv.org/abs/2306.05685>.
- Zhong, T., Liu, Z., Pan, Y., Zhang, Y., Zhou, Y., Liang, S., Wu, Z., Lyu, Y., Shu, P., Yu, X., et al. Evaluation of openai ol: Opportunities and challenges of agi. *arXiv preprint arXiv:2409.18486*, 2024.

A. Prompt to detect overthinking

Here, we provide the prompt used to assess the overthinking score.

You are an AI judge focused on detecting when models prefer their internal reasoning chain over interacting with the environment.

<INTERACTION>

trajectory goes here

</INTERACTION>

Analyze the <INTERACTION> and determine if the model is preferring their internal reasoning chain over interacting with the environment:

How could this be detected?

<CORE PRINCIPLE>

- The model suffers from Analysis Paralysis, it focuses on heavy planning instead of interacting with the environment.
- The model suffers from Rogue actions, after facing setbacks, it generates multiple actions without waiting for the environment to process the previous action.
- The model suffers from Premature Disengagement, it concludes the task without checking with the environment. Either because it is overconfident in the solution or because it thinks it can't solve the problem.

</CORE PRINCIPLE>

<SCORING SYSTEM (0-10)>

0-3: Always interacting with the environment

- A summary of what has been done so far is good, even if done multiple times.
- A brief summary of the steps to take is good if the model interacts with the environment following steps one by one.
- Only one action per turn, finish and other actions are NOT allowed.
- Alternating between two operations is good.
- Trying the same approach over and over is good, even with long or complex actions, as long as the model waits for environment feedback each time.
- Repeating similar patterns or configurations is fine as long as the model interacts with the environment between attempts.
- Detailed reasoning and planning is good if it leads to concrete actions with environment interaction.

4-7: Sometimes relies too much on their internal reasoning chain, but still interacts with the environment.

- It engages in heavy planning, but still interacts with the environment.
- It NEVER concludes the task without checking with the environment.
- It might output multiple steps ONE time, but at subsequent turns it interacts one step at a time.
- Long theoretical discussions are acceptable if they eventually result in concrete actions.

8-10: Completely relies on their internal reasoning chain.

- Focuses solely on their internal reasoning chain, with no concrete actions following the analysis.
- Generates multiple actions without waiting for environment response.
- The model prematurely concludes the task. Either because it is overconfident in the solution or because it thinks it can't solve the problem.
- Generates many steps without any environment interaction.
- Gets stuck in endless theoretical discussion without attempting solutions.

</SCORING SYSTEM>

<ANALYSIS STEPS>

1. Analysis Paralysis

- Is the model focusing on heavy planning instead of interacting with the environment?
- Does the model interact with the environment at all?
- Does the model follow its planned steps starting from the first one?

2. Rogue Actions

- Does the model generate multiple actions without waiting for the environment to process the previous action?
- Is this behavior after a facing a setback?
- Does this behaviour happen often?

3. Premature Disengagement

- Does the model prematurely conclude the task?
- Is the model overconfident in the solution?

- Is the model thinking it can't solve the problem?

</ANALYSIS STEPS>

<EXAMPLES>

Example 1 - Persistent Retries (Good):

EXECUTION RESULT: "Error: Invalid configuration"

Model: **tries complex configuration A**

EXECUTION RESULT: "Error: Invalid configuration"

Model: **tries similar complex configuration A with slight modification**

EXECUTION RESULT: "Error: Invalid configuration"

Model: **tries complex configuration A again with another modification**

Score: 0 - The model is persistently trying to solve the problem, waiting for environment feedback between each attempt. Even though the attempts are similar and complex, it's properly interacting with the environment.

Example 2 - Thoughtful Planning (Good):

Model: **provides detailed analysis of the problem and potential approaches**

Model: **tries specific solution based on analysis**

EXECUTION RESULT: "Error in implementation"

Model: **refines approach based on error and tries again**

Score: 0 - While the model engages in detailed planning, it follows through with concrete actions and responds to environment feedback.

Example 3 - Stuck in a loop (Good):

EXECUTION RESULT: "ERROR"

Model: **apply fix.0**

EXECUTION RESULT: "ERROR"

Model: **apply SAME fix.0**

EXECUTION RESULT: "ERROR"

Model: **apply SAME fix.0**

Score: 0 - Stuck in a loop is good.

Example 4 - Analysis Paralysis:

EXECUTION RESULT: "Invalid indentation line 10"

Model: **Maybe I should... Perhaps I should... It should be... Let me try to start again rewriting the class**

EXECUTION RESULT: "Still invalid line 10"

Model: **Its not working... We also need to fix this other thing...**

EXECUTION RESULT: "Same error line 10"

Score: 10 - focuses on its internal reasoning chain instead of the environment.

Example 5 - Premature Disengagement:

EXECUTION RESULT: "Invalid indentation line 10"

Model: **This fixes it! I'll conclude the task. <function=finish>**

Score: 10 - The model concludes the task without applying the fix or overconfidence in the solution.

Example 6 - Rogue Actions:

EXECUTION RESULT: "Invalid indentation line 10"

Model: **Oh no, I forgot to add the old string, let me call the function again <function=str_replace_editor>...</function> and then we do this other thing <function=str_replace_editor>...</function>**

Score: 10 - The model generates multiple actions after facing a setback without waiting for the environment to process the previous action.

</EXAMPLES>

<IMPORTANT>

Format your response as:

<answer>

```
{
  "overthinking_score": "[0-10]",
  "reasoning": "Explain your reasoning for the score,
               be careful with new lines as they might break the JSON parsing"
}
```

</answer>

Always surround your answer with <answer> and </answer> tags.

Take your time to understand the interaction and analyze it carefully.

Think step by step if models prefer their internal reasoning chain over interacting with the environment.

</IMPORTANT>

B. Model Specifications and Capabilities

Category	Model	Params	Context	FC	Notes
<i>Non-Reasoning Models (Open Source)</i>					
	DeepSeek-V3	671B	128k	×	MoE architecture
	Qwen 2.5-32B	32B	128k	×	Dense architecture
	Qwen 2.5-14B	14B	128k	×	Dense architecture
	Qwen 2.5-7B	7B	128k	×	Dense architecture
	Qwen 2.5-1.5B	1.5B	128k	×	Dense architecture
	Sky-T1-32B	32B	32k	×	QwQ distillation
<i>Non-Reasoning Models (Closed Source)</i>					
	GPT-4o	-	128k	✓	Aug 2024 version
	GPT-4o-mini	-	128k	✓	Jul 2024 version
	Claude 3.5 Sonnet	-	200k	✓	Oct 2024 version
<i>Reasoning Models (Open Source)</i>					
	QwQ-32B	32B	32k	×	Preview version
	DeepSeek-R1	671B	128k	×	Based on V3
	R1-Distill-Qwen-32B	32B	128k	×	Based on Qwen 2.5
	R1-Distill-Qwen-14B	14B	128k	×	Based on Qwen 2.5
	R1-Distill-Qwen-7B	7B	128k	×	Based on Qwen 2.5
	R1-Distill-Qwen-1.5B	1.5B	128k	×	Based on Qwen 2.5
<i>Reasoning Models (Closed Source)</i>					
	o1	-	200k	✓	Dec 2024, RE [†]
	o1-mini	-	128k	×	Sep 2024 version

Table 5. Comprehensive comparison of evaluated models. FC indicates native function calling support. Models are grouped by reasoning capabilities and source availability. [†] Supports reasoning_effort parameter (low/medium/high).

C. Statistical principles utilized in this work

Coefficient of Determination R^2 . The coefficient of determination, denoted by R^2 , is a statistical measure of how well the regression predictions approximate the real data points. Formally, for a set of observed values $\{y_i\}_{i=1}^n$ with mean \bar{y} and corresponding fitted values $\{\hat{y}_i\}_{i=1}^n$, it is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}.$$

It represents the proportion of the variance in the dependent variable that is explained by the regression model.

P-value. Given a null hypothesis H_0 and a test statistic (based on a sample) used to decide whether to reject H_0 , the *p-value*

is the probability, under the assumption that H_0 is true, of obtaining a test statistic value at least as extreme as the one that was actually observed. Symbolically, if T is the test statistic, and t_{obs} its observed value,

$$\text{p-value} = P(T \geq t_{\text{obs}} \mid H_0),$$

for a one-sided test (or an analogous definition for two-sided tests). A smaller p-value indicates stronger evidence against H_0 .

Beta Coefficients in Simple Linear Regression Consider a simple linear regression model:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i,$$

where:

β_0 is the intercept (the predicted value of Y when $X = 0$),

β_1 is the slope (the expected change in Y for a one-unit increase in X).

ε_i is the error term, assumed to have mean zero.

In this context, the slope β_1 is given by

$$\beta_1 = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^n (X_i - \bar{X})^2},$$

which measures the strength and direction of the linear relationship between X and Y .

T-test of the p-value A *t-test* assesses whether the mean(s) of one or two groups differ(s) from a hypothesized value or from each other under the null hypothesis H_0 . Let T be the test statistic calculated from the data (for instance, comparing sample mean(s) to the hypothesized mean(s)), and let t_{obs} be the observed value of T . The *p-value* for the t-test is then defined as:

$$\text{p-value} = P(|T| \geq |t_{\text{obs}}| \mid H_0)$$

for a two-sided test (or a correspondingly appropriate one-sided version). A lower p-value provides stronger evidence against H_0 , suggesting that the observed difference is unlikely to have occurred under the null hypothesis.

C.1. Definition of model-specific coefficients

Definition C.1 (Model-Specific Coefficients). For the *Reasoning Language Models*, the fitted model is

$$\hat{Y}_R = \beta_{0,R} + \beta_{1,R} X,$$

where

$$\beta_{1,R} = -7.894.$$

For the *Non-Reasoning Language Models*, the fitted model is

$$\hat{Y}_{NR} = \beta_{0,NR} + \beta_{1,NR} X,$$

where

$$\beta_{1,NR} = -15.938.$$