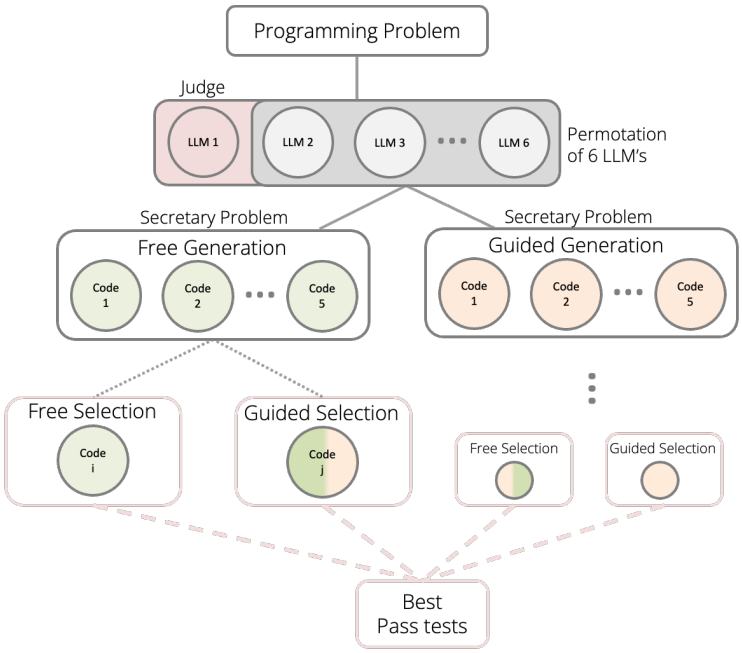
Freedom vs Constraints: Investigating Autonomy in LLM Code Evaluation through Cognitive Science Principles

Shaked Goldstein Technion
Niv Arad Technion
Assaf Schuster Technion

Abstract

Research in cognitive science has shown that humans often achieve better outcomes when granted greater freedom of choice, with autonomy linked to improved motivation and decision quality[1]. This study investigates whether a similar principle applies to large language models (LLMs) when evaluating and generating code. We compare two prompting conditions: a structured mode, where the model is instructed to select/generate code snippets based on explicit criteria (e.g., time or memory efficiency), and a freedom mode, where the model chooses or creates the "best" snippet without imposed constraints. To test this, we use a curated set of programming problems spanning multiple difficulty levels, drawn from the publicly available benchmark "APPS", supplemented with custom tasks designed to introduce meaningful trade-offs. Model performance was evaluated primarily through automated test-case execution, using pass rate as the main metric. Preliminary findings suggest that, much like the human mind, granting LLMs greater freedom of choice during code generation leads to more robust, adaptable, and higher-quality solutions, offering insight into whether LLMs exhibit human-like benefits from autonomy in decision-making and what this implies for prompt engineering and AI alignment.



1. Introduction

LLMs have rapidly advanced the field of code generation and evaluation, enabling systems that can solve programming problems, optimize algorithms, and assist developers in real time. Yet while significant research has focused on improving accuracy and efficiency of these models, less attention has been given to how decision autonomy, the degree of freedom granted to the model when selecting or generating solutions, affects performance. This question is particularly compelling when viewed through the lens of human cognition, where decades of psychological and neuroscientific research have shown that freedom of choice enhances motivation, engagement, and the quality of decisions. Studies such as *"Value of Freedom to Choose Encoded by the Human Brain"* demonstrate that the act of making autonomous choices activates reward-related brain regions, correlating with improved outcomes in humans. Similarly, work in LLM reasoning, including *"Tree of Thoughts: Deliberate Problem Solving with Large Language Models*[2]*,"* provides insight into how these models may emulate aspects of deliberate human thought. The central question of this study, therefore, is whether LLMs benefit from autonomy in a manner analogous to humans when making coding decisions.

To explore this question, we build on insights from Evaluating LLMs Trained on Code (Chen et al., 2021) and related benchmarks like HumanEval and APPS, adapting their methodologies to test freedom vs constraints. Unlike prior work that measures raw code-generation performance, our study systematically manipulates the freedom level given to LLMs in both generation (creating code) and selection (choosing among candidate snippets). This yields a two-dimensional framework: free vs guided generation crossed with free vs guided selection, allowing us to disentangle where autonomy helps or hinders performance.

Our experimental design spans three difficulty levels (easy, medium, hard) with approximately 50 problems per level, drawing from publicly available benchmarks and supplementing with custom tasks where trade-offs (e.g., speed vs readability, library use vs algorithmic implementation) are more pronounced. We evaluate multiple state-of-the-art models, including GPT, Command-A, Mistral, Claude and DeepSeek, across these conditions. Performance is assessed primarily through correctness, where for each chosen code, we check the pass rate by calculating the number of correct outputs divided by the total number of test cases.

This work contributes in two ways. First, it bridges cognitive science and AI, probing whether autonomy, a factor known to benefit human decision-making, transfers to machine learning models. Second, it offers practical insights for prompt engineering and multi-LLM systems: understanding when to constrain a model versus when to grant freedom could guide the design of hybrid code-assistant frameworks that dynamically adjust autonomy for optimal results.

2. Methodology

2.1 System Overview

The system developed for this project is based on a central agent that manages six different LLMs simultaneously. The agent receives as input a programming question drawn from a dataset containing three groups of problems at varying difficulty levels: easy, medium, and hard. Upon receiving a question, the agent randomly selects one of the models to act as the judge model, while the remaining models function as code generators.

To optimize the number of model calls, randomize the models generating the code, and simulate an efficient decision-making process, we implemented a strategy inspired by the Secretary Problem. Specifically, the agent first requests approximately $\frac{1}{e}$ of the models to generate code solutions in parallel. The judge model then compares these initial solutions in pairs, selecting the most promising one. The remaining models subsequently generate their solutions individually, and the first solution that surpasses the best from the initial subset is chosen as the final preferred solution. This mechanism was designed to reflect human-like evaluation efficiency, balancing exploration and exploitation when assessing alternatives.

2.2 Experimental Design

The experiment was divided into four distinct branches, each representing a specific combination of freedom and constraint in both code creation and solution selection:

easy_easy – free creation and free selection

easy_hard – free creation with guided (categorical) selection

hard_easy – guided creation with free selection

hard_hard – guided creation and guided selection

Each branch corresponds to a condition within a 2×2 autonomy framework, distinguishing between free and guided modes along both axes.

In the free prompt, models were instructed:

"**Write code that answers the question in the best possible way.**"

In the guided prompt, models were instructed:

"**Write code that answers the question in the best possible way, with emphasis on time and space complexity, code readability, abstraction level, and overall efficiency.**"

Each model thus operated under a specific level of autonomy, allowing for a controlled comparison between freedom and constraint conditions.

2.3 Dataset

To ensure diversity and reproducibility, we selected our programming questions from the APPS dataset (Hendrycks et al., 2021), a well-established benchmark for evaluating LLM performance on code generation tasks. The dataset contains a broad range of algorithmic and implementation-based problems, each accompanied by structured input–output test cases. For our experiment, we curated a balanced subset of 150 problems in total, consisting of:

50 easy-level problems (Introduction)

50 medium-level problems (Interview)

50 hard-level problems (Competition)

Each problem includes detailed specifications, constraints, and public test cases to facilitate consistent automated evaluation.

2.4 Evaluation Procedure

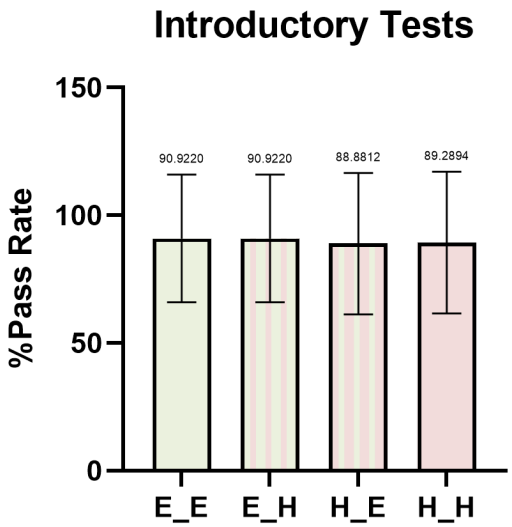
After completing the creation and selection phases for each branch, four final code snippets were obtained, one per experimental condition. These outputs were tested using an automated evaluation suite that executed the generated code against official APPS test cases.

The primary performance metric was functional correctness, expressed as the pass rate, the proportion of test cases for which the model's output matched the expected result. Additional

secondary metrics, such as execution time, memory consumption, and code readability, were recorded for future qualitative analysis.

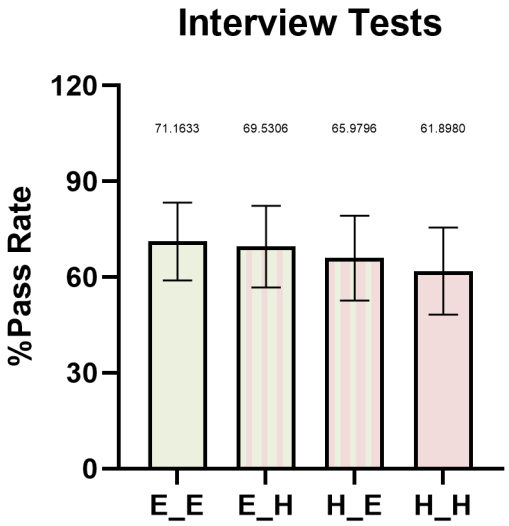
3. Results

Introductory Level



- Number of questions: 49
- Average success rates:
  - EASY_EASY – 90.9%
  - EASY_HARD – 90.9%
  - HARD_EASY – 88.9%
  - HARD_HARD – 89.3%
- Best branch: EASY_EASY
- Best-performing model: Claude 3.5 Sonnet (average 100%, SD 0.00)
- Other models: Deepseek 92.1%, Command-A 89.6%, Mistral 89.3%, Gemini 88.4%, GPT-4 83.1%
- Overall success rate across all questions: ~90%

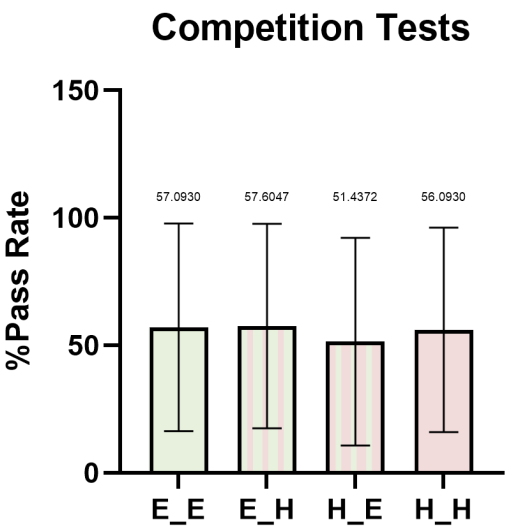
Interview Level



- Number of questions: 49
- Average success rates:
  - EASY_EASY – 71.2%
  - EASY_HARD – 69.5%
  - HARD_EASY – 66.0%
  - HARD_HARD – 61.9%
- Best branch: **EASY_EASY**
- Best-performing model: **Gemini 2.5 Flash** (average 84.4%, SD 0.37)
- Other models: Deepseek 73.5%, Claude 70.6%, Mistral 68.2%, Command-A 63.8%, GPT-4 36.3%
- Overall success rate across all questions: ~68%

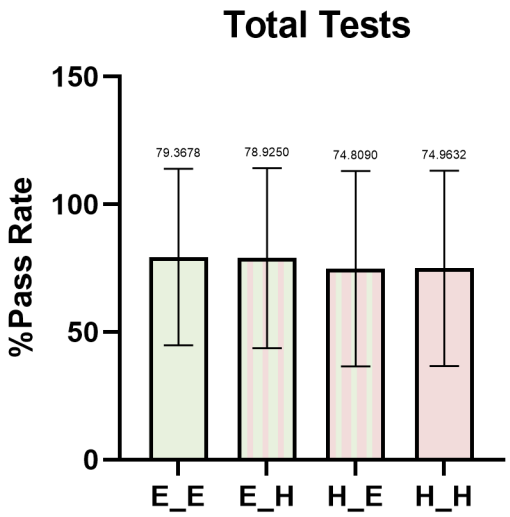
Competition Level

- Number of questions: 43
- Average success rates:
  - EASY_EASY – 57.1%
  - EASY_HARD – 57.6%
  - HARD_EASY – 51.4%
  - HARD_HARD – 56.1%
- Best branch: **EASY_HARD**
- Best-performing model: **Gemini 2.5 Flash** (average 83.9%, SD 0.28)
- Other models: Claude 47.6%, Mistral 47.0%, Command-A 40.1%, Deepseek 33.5%, GPT-4 22.3%
- Overall success rate across all questions: ~55%



General Summary

The **EASY_EASY** branch achieved the best results in the Introductory and Interview levels, while **EASY_HARD** performed best in the Competition level, indicating that full freedom (in creation and selection) benefits simpler tasks, whereas partial guidance leads to better performance in complex ones.



4. Discussion

The results of this study demonstrate that the degree of autonomy granted to LLMs meaningfully influences their performance in code generation tasks, in a way that parallels human cognitive behavior. Across all difficulty levels, the *freedom mode*, where models were allowed to generate and select solutions without strict constraints yielded higher success rates on simpler tasks, while *guided modes* provided advantages as task complexity increased. Specifically, the EASY_EASY branch (free creation and free selection) achieved the highest accuracy in the Introductory and Interview levels, suggesting that open-ended decision-making enables LLMs to explore a broader solution space and leverage implicit reasoning patterns. In contrast, at the Competition level, EASY_HARD (free creation with guided selection) performed best, implying that partial structure during evaluation helps refine model output when problem-solving demands become more intricate.

These findings align with well-established theories in cognitive science regarding autonomy and performance. Studies such as *"Value of Freedom to Choose Encoded by the Human Brain"* show that human motivation and decision quality improve when individuals are allowed to exercise freedom of choice. Analogously, our results suggest that when LLMs are not over-constrained by explicit instructions, they may exhibit more diverse internal reasoning paths, producing creative, robust, and well-generalized solutions. However, as task complexity increases, unguided autonomy can lead to inefficiencies similar to human over-exploration, reinforcing the need for a balance between freedom and structure.

From an AI engineering perspective, these results have practical implications for prompt design and multi-agent orchestration. In systems that rely on multiple models or iterative reasoning (such as the one developed for this project), adjusting the level of freedom dynamically according to problem complexity may lead to more optimal outcomes. For example, allowing unconstrained reasoning in early stages and introducing structured evaluation criteria later mirrors both the Secretary Problem approach and human decision strategies that balance exploration and exploitation. This adaptive autonomy mechanism could inform the design of future hybrid code-generation frameworks and self-reflective model systems.

Nonetheless, this study has several limitations. The dataset, while diverse, was restricted to 150 problems from the APPS benchmark, which may not capture the full variability of real-world

coding scenarios. Moreover, the evaluation primarily focused on correctness, with secondary factors such as readability and abstraction analyzed qualitatively rather than quantitatively. Future research should incorporate larger datasets, more granular behavioral metrics, and longitudinal comparisons across model updates to better understand how autonomy interacts with training architecture and reasoning depth.

Overall, these results provide empirical evidence that autonomy can enhance LLM performance in code generation in a manner reminiscent of human decision-making patterns. However, the observed shift, where moderate guidance becomes advantageous as complexity rises, suggests that *optimal autonomy* for LLMs is task-dependent. This insight highlights a critical opportunity for adaptive prompt engineering and for drawing deeper parallels between human and artificial decision-making systems.

5. References

1. Fujiwara, J. *et al.* Value of freedom to choose encoded by the human brain. *J. Neurophysiol.* **110**, 1915–1929 (2013).

2. Yao, S. *et al.* Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *arXiv* (2023) doi:10.48550/arxiv.2305.10601.

3. Chu, Z. *et al.* Navigate through Enigmatic Labyrinth A Survey of Chain of Thought Reasoning: Advances, Frontiers and Future. in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* 1173–1203 (Association for Computational Linguistics, 2024). doi:10.18653/v1/2024.acl-long.65.