Do LLMs Scale on Inductive Proofs? A Controlled Study of Elementary Arithmetical Proofs

Risako Ando
Department of Philosophy, Keio University, Japan

Koji Mineshima
Department of Philosophy, Keio University, Japan

Mitsuhiro Okada
Department of Philosophy, Keio University, Japan

1 Introduction

Large language models (LLMs) have recently made remarkable progress in mathematical reasoning and formal theorem proving. However, it remains unclear to what extent these achievements reflect the intrinsic proof-construction ability of LLMs rather than reliance on theorem retrieval, external libraries such as Lean's Mathlib, and automated tactics [3].

To investigate this question, we study theorem proving in a controlled *proof-from-scratch* setting, where models cannot rely on pre-established lemmas, libraries, or tactics. Building on our previous work [1], we evaluate LLMs' performance on direct inductive proofs, including proofs requiring nested induction. For comparison, we also evaluate algebraic proofs in elementary arithmetic. To assess proof construction across varying levels of complexity, we also evaluate models on commutativity theorems formalized and verified in Lean.

2 Problem Setting and Experiment Design

We evaluate LLMs on the following proof styles over elementary arithmetic.

- *Direct inductive proofs* are proofs constructed using only mathematical induction over the natural numbers together with the recursive definitions of addition and multiplication. When required, proofs must explicitly employ nested induction.

All direct inductive proofs can be formalized within a restricted fragment of quantifier-free Primitive Recursive Arithmetic (PRA) [5]. Although the target domain is limited to a simple fragment of PRA, nested induction still gives rise to nontrivial proof structures and requires careful management of induction hypotheses [2].

We also compare the following styles of proofs:

- *Inductive proofs with lemmas* allow auxiliary lemmas, provided each lemma is accompanied by its proof.
- *Algebraic proofs* are proofs constructed using only the recursive definitions of addition and multiplication together with the five fundamental algebraic laws of arithmetic: the commutativity and associativity of addition and multiplication, and the distributivity of multiplication over addition.

Models are instructed to construct formal proofs in Lean using few-shot prompting. To examine models' behavior without any detailed instructions or examples, we also test a *zero-shot* prompt where models are simply asked "Prove by induction" without feeding any example. In all settings, the use of Lean's Mathlib library and automated tactics is prohibited.

As a baseline set, we use the 20 arithmetic problems from our previous benchmark [1] (e.g., $a + 1 = 1 + a$). In addition, we introduce a new evaluation set designed to probe the limits of LLMs' formal proof generation capabilities in a controlled setting. The new dataset consists of 20 commutativity theorems for addition generated by progressively increasing both the number of variables and the number of required applications of the commutativity

42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:3
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

law. The problems range from $(a+b)+c=c+(b+a)$ and $((a+b)+c)+d=d+(c+(b+a))$ to formulas containing up to 22 variables.

We evaluate LLMs across varying levels of proof complexity. For direct inductive proofs, complexity is measured by the maximum nesting depth of mathematical induction. For algebraic proofs, complexity is measured by the number of applications of algebraic laws.

We evaluate a state-of-the-art reasoning model (GPT-5.2) and a standard model (GPT-4o) on Lean 4 formal proof generation tasks.

3 Experimental Results and Discussion

The results on the baseline set are as follows. For direct inductive proofs, GPT-5.2 solved 10/20 problems, substantially outperforming GPT-4o, which solved 6/20. For inductive proofs with lemmas, GPT-5.2 again substantially outperformed GPT-4o, solving 18/20 problems compared with 7/20 for GPT-4o. For algebraic proofs, GPT-5.2 solved 19/20 baseline problems, compared with 13/20 for GPT-4o. Overall, GPT-5.2 consistently outperformed GPT-4o across all proof styles. Moreover, constructing direct inductive proofs proved substantially more difficult than lemma-based or algebraic proofs.

When simply instructed to "prove by induction" without specifying a proof style or providing examples, GPT-5.2 solved 16/20 problems, but only 4 were direct inductive proofs; the remaining 12 relied on auxiliary algebraic lemmas. In contrast, GPT-4o solved 4/20 problems, all by direct induction.

On the commutativity set, performance dropped markedly. GPT-5.2 solved 6/20 direct inductive proofs, 8/20 inductive proofs with lemmas, and 7/20 algebraic proofs. GPT-4o failed on all direct inductive and lemma-based proofs, while solving 8/20 algebraic proofs.

For direct inductive proofs, GPT-5.2 succeeded only on the six simplest instances (induction depths 3–8), indicating a clear complexity threshold. Most failures are due to Lean syntax errors rather than incorrect proof steps.

For algebraic proofs, both models performed well only on low-complexity instances. GPT-5.2 succeeded on the first seven problems, requiring 2–8 applications of algebraic laws, but failed on all remaining problems, again almost exclusively because of Lean syntax errors. GPT-4o achieved a slightly higher score (8/20), but its performance is less systematic: it fails on several low-complexity instances while solving some more difficult ones.

Qualitative analysis shows that the dominant failure mode is syntactic errors, primarily mismatched parentheses in Lean 4 scripts. This finding is consistent with previous work showing that Transformer-based models struggle to maintain syntactic well-formedness in structurally complex formal languages such as Dyck languages [4].

We further analyzed the complexity of the generated algebraic proofs. For low-complexity problems, both models tend to produce unnecessarily complex proofs by introducing redundant applications of algebraic laws. In particular, although the first two problems have target complexities of 2 and 3, GPT-5.2 generates proofs of complexities 6 and 7, respectively, while GPT-4o generates a proof of complexity 6 for the first problem. For all subsequent problems, the complexity of the generated proofs exactly matched the target complexity.

4 Conclusion and Ongoing Work

While this paper focuses on addition commutativity, preliminary experiments on multiplication suggest similar degradation for multiplication commutativity, associativity, and distributivity. Our ongoing work investigates whether informal natural-language proof generation mitigates this bottleneck and how deeply nested induction affects LLM reasoning.

References

1 Risako Ando, Koji Mineshima, and Mitsuhiro Okada. Direct induction proof challenge: Evaluating large language models on deeply nested mathematical induction. In *The Second AI for MATH Workshop at the 42nd International Conference on Machine Learning (ICML)*, Vancouver, Canada, 2025. URL: `https://openreview.net/pdf?id=kFrk1m2qSW`.

2 Robert S. Boyer and J Strother Moore. *A Computational Logic*. Academic Press, New York, 1979.

3 Walter Dean and Alberto Naibo. Artificial intelligence and inherent mathematical difficulty. *Philosophia Mathematica*, 33(3):283–329, 2025.

4 Michael Hahn. Theoretical limitations of self-attention in neural sequence models. *Transactions of the Association for Computational Linguistics*, 8:156–171, 2020.

5 Thoralf Skolem. The foundation of elementary arithmetic established by means of the recursive mode of thought, without the use of apparent variables ranging over infinite domains. In Jean van Heijenoort, editor, *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931*, pages 302–333. Harvard University Press, 1967. Original work published in 1923.