# Can Large Language Models Support Proving Theorems Involving Multiply Nested Mathematical Induction? — A Preliminary Report —

Risako Ando, Koji Mineshima, and Mitsuhiro Okada

Keio University, Tokyo, Japan

## 1 Introduction

It has become widely recognized in recent years that large language models (LLMs) possess a certain capacity for logical reasoning and its arithmetical extensions, including the ability to construct proofs. However, many of these successful instances might be attributable to the reuse and combination of lemmas contained in the training data. This corresponds to the use of pre-existing lemmas in libraries of automated theorem proving and proof assistant systems.

In this study, we focus on proof construction within the domain of elementary arithmetic, under conditions that restrict access to such lemmas. Specifically, we investigate the extent to which LLMs can construct proofs involving mathematical induction of varying complexity, using the depth of nested induction as a measure of complexity. Furthermore, we evaluate whether LLMs can generalize to more complex inductive proofs when given examples of simpler ones in a few-shot prompting setting. This research project is in its early stages, and we report the goals and current progress.

## 2 Direct induction proofs with multiply nested induction

Research on the automation of mathematical induction dates back to the 1970s [1], with foundational systems such as the Boyer-Moore theorem prover pioneering mechanized reasoning with induction. Modern proof assistants such as Coq and Lean provide robust support for inductive types and user-guided inductive reasoning. However, despite these advancements, the full automation of inductive proofs remains a significant challenge. In practice, users are still required to manually design induction schemas or supply key lemmas, and general-purpose automation often fails to scale beyond relatively simple proofs.

Meanwhile, the application of LLMs to mathematics has been rapidly progressing [6]. Dean and Naibo [2] analyzed the proof capabilities of current LLMs in detail by classifying logical formulas based on their complexity in the arithmetical hierarchy. In contrast, we focus specifically on mathematical induction—an area where theorem proving still faces significant challenges—and analyze LLMs' ability to construct inductive proofs.

LLMs tend to employ induction in a shallow manner, often relying on existing libraries of lemmas. To investigate how well LLMs can support proofs involving nested induction, we focus on what we call *direct induction proofs*. Suppose an inductive data type is given, along with a set of primitive recursive function definitions induced from it. When the structural induction principle is provided as an inference rule, we define a direct induction proof as a proof of a universally quantified equation that is constructed solely from the given definitions and the structural induction principle—that is, without invoking any auxiliary lemmas. We assume equational logic as the background logic.

Among various inductive data types, a fundamental case is that of the natural numbers. Skolem's [7] quantifier-free Primitive Recursive Arithmetic (PRA) corresponds to this setting, where the inductive type is the natural numbers. For the purposes of this study, we primarily focus on a fragment of PRA in which addition and multiplication are defined directly. This fragment is sufficiently rich in that it allows multiply nested induction, making it a valuable first step for examining the automation of direct induction proofs.

# 3 Pilot Experiments

**Data and Tasks:** In this preliminary report, we constructed a set of arithmetic statements involving addition and multiplication. Specifically, we created 20 problems in total, with 5 examples each containing between one and four variables. For each problem, we instructed LLMs to generate two types of proofs: an informal proof in natural language and a formal proof written in Lean 4 (`https://lean-lang.org/`). We then manually examined the correctness of two types of proofs.

Previous work [5] has shown that in basic logical reasoning tasks such as syllogisms, prompting LLMs to produce logical formulas followed by explanations based on them can improve accuracy. We aim to test whether a similar effect can be observed in mathematical proof construction. In addition, generating Lean code makes it easier to automatically verify the correctness of proofs using the proof assistant itself.

**Experimental Setting:** In this experiment, we used OpenAI's gpt-3.5-turbo and gpt-4o [4, 3] via the API, as a baseline and a more advanced model, respectively. We set the temperature to 0 and the maximum number of output tokens to 2000, while leaving other parameters at their default values.

In the prompt for Lean proofs, we instructed the model to follow these constraints: (1) do not use any predefined lemmas; (2) use only mathematical induction on natural numbers, including nested induction if necessary; and (3) do not use any automated tactics. For the prompt requesting informal proofs, we gave the models similar instructions.

We also gave two-shot examples: one for a proof of $a + succ(0) = succ(a)$, and another involving double induction for the statement $a + b = b + a$, along with the definitions of natural numbers, addition, and multiplication.

**Results and Analysis:** As an initial result, the following observations were made: With gpt-4o, for informal proofs, correct proofs were obtained for 7 out of 20 problems. For formal proofs, only 1 correct proof was produced. With gpt-3.5, correct proofs were generated for 3 problems in the informal proof setting, and only 1 in the formal proof setting.

Regarding the generalization abilities of LLMs in theorem proving, the following results were observed: (1) Despite not being given any examples involving multiplication, there were cases in which the model constructed correct direct induction proofs involving multiplication. (2) Although only examples of double induction were provided, there were instances in the formal proofs where the model attempted to use triple or deeper nested induction. However, these proofs were not correct. (3) A common error pattern was the failure to adhere to the provided definitions, with many proofs including unproven steps such as $succ(a) + 0 = succ(a)$, even though the definition is $0 + succ(a) = succ(a)$. (4) Additionally, many proofs were found to be correct but not direct induction proofs, as they relied on auxiliary lemmas.

**Future Work:** We plan to provide Lean error messages as feedback to the LLM, enabling it to iteratively refine its proofs. We will also conduct evaluations with automated theorem provers that support mathematical induction. Furthermore, we aim to extend our experiments within the framework of PRA to encompass functions beyond addition and multiplication. Finally,

we intend to broaden our investigation to include more general inductive data types, allowing for an evaluation of mathematical induction in the context of more general forms of structural induction.

# References


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

[2] Walter Dean and Alberto Naibo. Artificial intelligence and inherent mathematical difficulty. *arXiv preprint arXiv:2408.03345*, 2024.

[3] OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[4] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[5] Kentaro Ozeki, Risako Ando, Takanobu Morishita, Hirohiko Abe, Koji Mineshima, and Mitsuhiro Okada. Exploring reasoning biases in large language models through syllogism: Insights from the NeuBAROCO dataset. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 16063–16077, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[6] Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

[7] 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.