Autoformalization and autoinformalization via probabilistic reasoning

Cameron E. Freer[1], Alexander K. Lew[1,2],
Timothy J. O'Donnell[3], and Vikash K. Mansinghka[1]

[1] MIT, Cambridge, MA, USA {freer, alexlew, vkm}@mit.edu
[2] Yale University, New Haven, CT, USA
[3] McGill University, Mila – Québec AI Institute, Montréal, QC, Canada
timothy.odonnell@mcgill.ca

Abstract. Recent advances in the control of large language models via probabilistic programming allow us to imagine new workflows in theorem proving and formal mathematics. In order to spark a broader conversation about the role of probabilistic reasoning, we propose several speculative approaches: a *reweighted wake-sleep* algorithm for improving model quality, an *involutive* approach to autoformalization and autoinformalization, and a *coarse-to-fine* approach to proof synthesis.

The *controlled generation* of large language model (LLM) output has been successfully framed in terms of a probabilistic inference problem, specifically the task of sampling from the posterior distributions of an LLM conditioned on various constraints. These constraints can be hard (e.g., ensuring code adheres to a formal grammar, or that a proof is verified by a theorem prover) or soft (e.g., accuracy as scored by another LLM, or likelihoods from a probabilistic model with domain knowledge). By using conditional probabilities of LLM completions (see Figure 1), probabilistic programming and sequential Monte Carlo (SMC) techniques have enabled systems like LLaMPL [8] and GenParse [9] that generate asymptotically correct samples under these diverse constraints.

Controlled generation via probabilistic reasoning opens the possibility of new workflows for *autoformalization* (translating natural-language theorems and proofs into statements that could be formally checked by a theorem prover) and *autoinformalization* (generating natural language explanations from formal theorem statements and proofs).

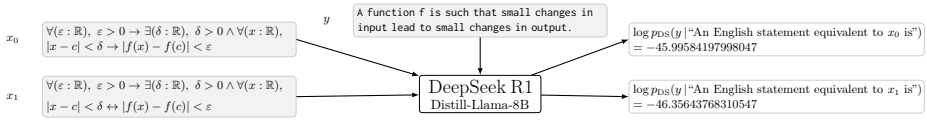

Fig. 1. *Conditional probabilities of LLM completions can identify better formalizations:* Two candidate formalizations x_0, x_1 in Lean are judged based on the conditional log probabilities assigned to an informal statement y by a distillation [1] of DeepSeek R1 [2]. In this case, $p_{\mathrm{DS}}(y|x_0) > p_{\mathrm{DS}}(y|x_1)$, identifying x_0 as a more correct formalization of y.

Understanding an informal proof involves inferring the author's intent regarding choice of definitions, domains of variables, implicit assumptions or transformations, and references or (counter)examples that fill in argument details. Such inferences exploit signals about the likely meaning of the proof, and draw on both hard constraints (like typechecking) and softer signals (such as avoiding interpretations that make a theorem trivial).

Our prior techniques [8, 9] allow augmenting and steering of LLM-guided code generation using probabilistic programs that condition LLM generation on such hard and soft signals. Although high-quality generation using these signals can be expensive, generated examples (or entire traces of the reasoning process induced by the SMC inference algorithm we use) can serve as training data, in a reinforcement-learning loop, to train models that have this sort of probabilistic reasoning "built in." When such signals are unavailable to the SMC proposal, they can still be used to guide inference via *twist functions* [9, 11].

Figure 2 illustrates three potential workflows for auto(in)formalization using probabilistically constrained LLMs, which we hope will spark further exploration of the use of probabilistic reasoning in theorem proving and formal mathematics. These workflows also suggest questions such as how fine-tuning based on constrained outputs (2a) affects scaling laws for pre- vs. post-training, how probabilistic teachers and learners (2b) can improve distillation, and how hierarchical probabilistic refinement (2c) can dovetail with chain-of-thought techniques.

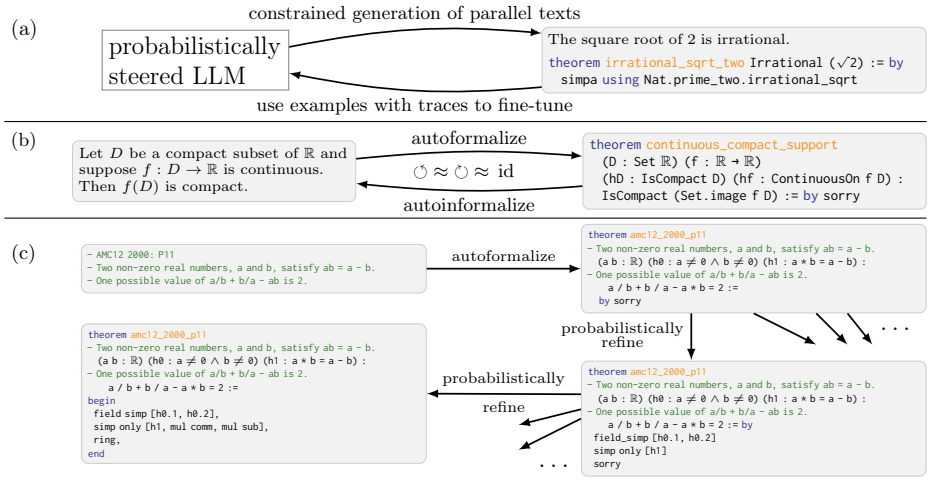

Fig. 2. *Three proposed workflows using probabilistic reasoning, illustrated with Lean code examples* [3, 6, 10, 12]: (a) **Reweighted wake-sleep algorithm:** A model can improve itself by searching during training for high-quality labeled examples that are then used for fine-tuning, better leveraging the signal provided by methods that use proof search results as training data [7]. (b) **Involutive approach to auto(in)formalization:** Using mutual recursion, informalization is framed as "writing English that an autoformalizer would likely translate correctly," and formalization as "writing Lean that an autoinformalizer would likely agree matches the informal statement." (c) **Coarse-to-fine proof synthesis:** Proof synthesis as an instance of Coarse-to-Fine probabilistic program induction [4, 5]; a proof's basic structure is first formalized at a coarse level with high entropy and then refined, guided by probabilistic inferences about its likely fidelity to a coarsening of the informal proof.

References

[1] DeepSeek AI. *DeepSeek-R1-Distill-Llama-8B*. https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-8B. 2025.

[2] DeepSeek AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, et al. *DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning*. 2025. arXiv: 2501.12948 [cs.CL]. URL: https://arxiv.org/abs/2501.12948.

[3] Zhangir Azerbayev, Bartosz Piotrowski, Hailey Schoelkopf, Edward W. Ayers, Dragomir Radev, and Jeremy Avigad. *ProofNet: Autoformalizing and Formally Proving Undergraduate-Level Mathematics*. 2023. arXiv: 2302.12433 [cs.CL].

[4] Maddy L. Bowers, Alexander K. Lew, Vikash K. Mansinghka, Joshua B. Tenenbaum, and Armando Solar-Lezama. "Toward Probabilistic Coarse-to-Fine Program Synthesis". In: *Languages for Inference (LAFI 2024)*. 2024. URL: https://popl24.sigplan.org/details/lafi-2024-papers/18/Toward-Probabilistic-Coarse-to-Fine-Program-Synthesis.

[5] Maddy L. Bowers, Alexander K. Lew, Wenhao Qi, Joshua S. Rule, Vikash K. Mansinghka, Joshua B. Tenenbaum, and Armando Solar-Lezama. "Concept Learning as Coarse-to-Fine Probabilistic Program Induction". In: *Proceedings of the Annual Meeting of the Cognitive Science Society (CogSci 2024)*. Vol. 46. 2024. URL: https://escholarship.org/uc/item/52t1w32c.

[6] The mathlib Community. "The Lean Mathematical Library". In: *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP 2020)*. 2020, pp. 367–381. DOI: 10.1145/3372885.3373824.

[7] Jesse Michael Han, Jason Rute, Yuhuai Wu, Edward Ayers, and Stanislas Polu. "Proof Artifact Co-Training for Theorem Proving with Language Models". In: *International Conference on Learning Representations (ICLR 2022)*. 2022. URL: https://openreview.net/forum?id=rpxJc9j04U.

[8] Alexander K. Lew, Tan Zhi-Xuan, Gabriel Grand, and Vikash K. Mansinghka. *Sequential Monte Carlo Steering of Large Language Models using Probabilistic Programs*. 2023. arXiv: 2306.03081 [cs.AI].

[9] João Loula, Benjamin LeBrun, Li Du, Ben Lipkin, Clemente Pasti, Gabriel Grand, Tianyu Liu, Yahya Emara, Marjorie Freedman, Jason Eisner, Ryan Cotterell, Vikash Mansinghka, Alexander K. Lew, Tim Vieira, and Timothy J. O'Donnell. "Syntactic and Semantic Control of Large Language Models via Sequential Monte Carlo". In: *International Conference on Learning Representations (ICLR 2025)*. 2025. URL: https://openreview.net/forum?id=xoXn62FzD0.

[10] Huaiyuan Ying, Zijian Wu, Yihan Geng, Jiayu Wang, Dahua Lin, and Kai Chen. *Lean Workbook: A large-scale Lean problem set formalized from natural language math problems*. 2024. arXiv: 2406.03847 [cs.CL].

[11] Stephen Zhao, Rob Brekelmans, Alireza Makhzani, and Roger Baker Grosse. "Probabilistic Inference in Language Models via Twisted Sequential Monte Carlo". In: *Proceedings of the 41st International Conference on Machine*

Learning (ICML 2024). Vol. 235. 2024, pp. 60704–60748. URL: https://proceedings.mlr.press/v235/zhao24c.html.

[12] Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. "miniF2F: a cross-system benchmark for formal Olympiad-level mathematics". In: *International Conference on Learning Representations (ICLR 2022)*. 2022. URL: https://openreview.net/forum?id=9ZPegFuFTFv.