NLIR: Natural Language Intermediate Representation for Mechanized Theorem Proving

Laetitia Teodorescu,[1] Guillaume Baudart,[2]
Emilio Jesús Gallego Arias,[2] Marc Lelarge[3]

[1] AdaptiveML
[2] IRIF, Université Paris Cité, Inria, CNRS
[3] DI ENS, PSL University, Inria

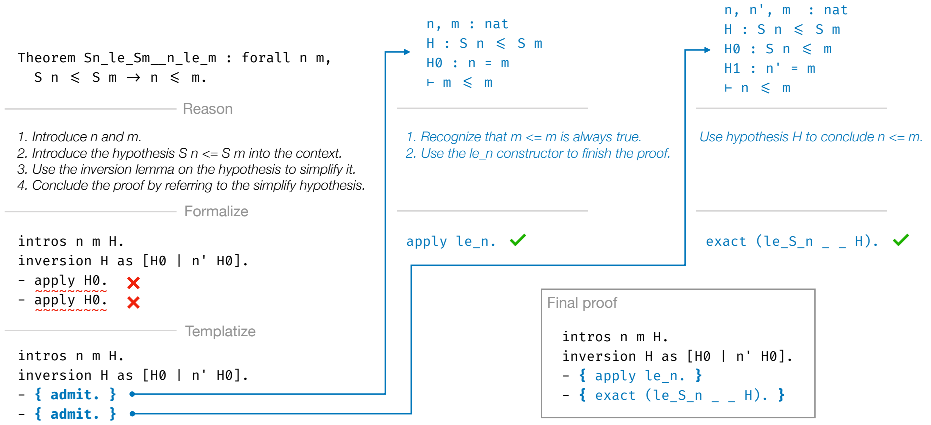
Thanks to recent advances in LLM capabilities, we believe natural language can serve as a universal interface for reasoning about formal proofs. Using natural language leverages the strength of LLMs, and allows us to use chain-of-thought [1] by asking for an informal mathematical proof before generating the formal proof, making it more intuitive and comprehensible compared to purely automatic formal techniques. Additionally, partial proofs expressed in natural language are easier for humans to understand, adapt, or reuse, allowing for greater flexibility and collaboration between machine-generated suggestions and human mathematicians.

In this work, we develop an LLM-based agent that can interact with the Rocq proof assistant. We present the following contributions: 1) *Pétanque*: A new fast and lightweight environment to interact with the Rocq theorem prover. 2) An interactive proof protocol leveraging natural language reasoning: hierarchical proof templating. 3) A search algorithms leveraging feedback from the ITP and natural language to rerank proof candidates.

Pétanque: a lightweight interactive environment for Rocq Following existing work [2,3,4,5], we have built a new environment for machine to machine interaction for the Rocq proof assistant, particularly tailored for interactive, high-throughput, low-latency learning applications. Pétanque is based on Flèche [6], a new document manager for Rocq.

Hierarchical proof templating The templating agent tries to generate full proofs directly. Failed tactics are replaced with *holes* to obtain a proof *template*. The sub-goal corresponding to each holes is then fed back to the agent which repeat the process to fill the holes one by one using focused fine-grain reasoning. The proof is complete when there are no more holes. Our approach's originality is that although the protocols' inputs (goals) and outputs (tactics) are Rocq code, the agent internally uses natural language as an intermediate representation to analyze the input and guide the code generation. An example execution of the hierarchical proof templating agent is presented in Figure 1.

Proof search We combine our interactive protocol with the classic beam search algorithm. Inspired by [7], we use the LLM to rank and sort the proposals during the search. At each step, the LLM generates n possible proofs. We use Pétanque to templatize each proof and store all the resulting templates. Then, the LLM discuss, compare and finally rank and sort the candidates for the next step.



```
Theorem Sn_le_Sm__n_le_m : forall n m,
  S n ⩽ S m → n ⩽ m.
```

1. Introduce n and m.
2. Introduce the hypothesis S n <= S m into the context.
3. Use the inversion lemma on the hypothesis to simplify it.
4. Conclude the proof by referring to the simplify hypothesis.

```
intros n m H.
inversion H as [H0 | n' H0].
- apply H0. ✗
- apply H0. ✗
```

```
intros n m H.
inversion H as [H0 | n' H0].
- { admit. }
- { admit. }
```

```
n, m : nat
H : S n ⩽ S m
H0 : n = m
⊢ m ⩽ m
```

1. Recognize that m <= m is always true.
2. Use the le_n constructor to finish the proof.

```
apply le_n. ✓
```

```
n, n', m  : nat
H : S n ⩽ S m
H0 : S n ⩽ m
H1 : n' = m
⊢ n ⩽ m
```

Use hypothesis H to conclude n <= m.

```
exact (le_S_n _ _ H). ✓
```

```
intros n m H.
inversion H as [H0 | n' H0].
- { apply le_n. }
- { exact (le_S_n _ _ H). }
```

Fig. 1. Hierarchical proof templating.

Evaluation To limit data leaks issues, we extracted the first 100 lemmas from the recent proof of $BB(4) = 107$ [8]. To provide the necessary context for the proof, for each lemma we augment the prompt with all the preceding definitions and lemmas. We evaluate our agent with 3 state-of-the-art LLMs: GPT-4o, LLaMa-3.3, and DeepSeek-v3. The number of proposal is 4, the beam search is 3, and the maximum number of search steps is 10. The gray numbers indicate the number of proofs that were correct at the first try (no holes).

	GPT-4o	LLaMa-3.3	DeepSeek-V3
% success	(38.0) 58.0	(20.0) 46.0	(30.0) 56.0

Related work Closest to our work, [9] build a tactic-by-tactic LLM agent based on GPT-4 and also use an interface to summarize past interactions. They, however, do not use proof repair or beam search. Also close to our work, [10] use proof repair over hierarchical proofs in Isabelle, coupled with best-first search. Contrary to us, they use fine-tuned models and no chain-of-thought. Finally, [11] propose a framework for training language models to produce informal thoughts prior to each step of a proof, thereby boosting the model's theorem-proving capabilities.

This work is also related to recent investigations on the reasoning abilities of LLMs [12]. Chain-of-Thought (CoT) prompting [1] was shown to improve LLM's answers; subsequent work found that these reasoning abilities could be elicited zero-shot [13]. Further work interleaved CoT with decision-making [14], added search and complex control flow to reasoning [15,7,16], incorporated refinement and feedback [17,18], and learned to generate novel reasoning traces that proved beneficial for further training [19,20]. Like our work, many of these methods – especially the ones using search and refinement – make use of LLM-based scoring or ranking functions [21].

A previous version of this work was presented at the MathAI@NeurIPS 2024 workshop [22].

References

1. Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.
2. Emilio Jesús Gallego Arias, Benoît Pin, and Pierre Jouvelot. jscoq: Towards hybrid theorem proving interfaces. In Serge Autexier and Pedro Quaresma, editors, *UITP*, 2016.
3. Emilio Jesús Gallego Arias. SerAPI: Machine-friendly, data-centric serialization for Coq. preprint, 01 2019.
4. Kaiyu Yang and Jia Deng. Learning to prove theorems via interacting with proof assistants. In *ICML*, 2019.
5. Alex Sanchez-Stern, Yousef Alhessi, Lawrence K. Saul, and Sorin Lerner. Generating correctness proofs with neural networks. In *MAPL@PLDI*, 2020.
6. Emilio Jesús Gallego Arias. Flèche: Incremental validation for hybrid formal documents. under revision, 2024.
7. Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*, 2023.
8. ccz181078. https://github.com/ccz181078/Coq-BB5/tree/main, 2024.
9. Amitayush Thakur, George D. Tsoukalas, Yeming Wen, Jimmy Xin, and Swarat Chaudhuri. An in-context learning agent for formal theorem-proving. In *COLM*, 2024.
10. Haiming Wang, Huajian Xin, Zhengying Liu, Wenda Li, Yinya Huang, Jianqiao Lu, Zhicheng Yang, Jing Tang, Jian Yin, Zhenguo Li, and Xiaodan Liang. Proving theorems recursively. *CoRR*, abs/2405.14414, 2024.
11. Haohan Lin, Zhiqing Sun, Yiming Yang, and Sean Welleck. Lean-star: Learning to interleave thinking and proving. *arXiv preprint arXiv:2407.10040*, 2024.
12. Aske Plaat, Annie Wong, Suzan Verberne, Joost Broekens, Niki van Stein, and Thomas Back. Reasoning with large language models, a survey. *arXiv preprint arXiv:2407.11511*, 2024.
13. Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *NeurIPS*, 2022.
14. Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.
15. Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.
16. Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *AAAI*, 2024.
17. Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *NeurIPS*, 2024.
18. Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *NeurIPS*, 2024.

19. Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *NeurIPS*, 2022.
20. Eric Zelikman, Georges Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah D Goodman. Quiet-star: Language models can teach themselves to think before speaking. *arXiv preprint arXiv:2403.09629*, 2024.
21. Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *NeurIPS*, 2023.
22. Laetitia Teodorescu, Guillaume Baudart, Emilio Jesús Gallego Arias, and Marc Lelarge. Nlir: Natural language intermediate representation for mechanized theorem proving. In *MathAI@NeurIPS*, 2024.