Generalization of LLMs in SAT Reasoning via Structured Scratchpad Interaction

František Koutenský[1], Petr Hyner[2], and Jan Hůla[1, 2]

[1]Czech Technical University in Prague
[2]University of Ostrava

1 Introduction

Large Language Models (LLMs) have recently demonstrated promising multi-step reasoning capabilities through techniques like chain-of-thought prompting and scratchpads. However, they struggle with tasks requiring complex search and backtracking, such as SAT solving. Our goal is to examine whether LLMs can learn and generalize the Conflict-Driven Clause Learning (CDCL) algorithm from supervised solver traces.

While prior work [4, 3] showed that graph neural networks (GNNs) can learn SAT-solving behavior from a satisfiability signal alone, LLMs operate on sequential text and require different inductive biases and representations. SatLM [6] proposes offloading logical reasoning to external SAT solvers by generating declarative specifications, ensuring correctness at the cost of end-to-end learning. In contrast, we embed the CDCL procedure directly into the LLM, allowing it to learn and execute the solving process without relying on symbolic backends.

Our work is motivated by recent advances in neural algorithmic reasoning [1], which show that models can learn algorithmic tasks and generalize out-of-distribution (OOD) when trained properly. Additionally, agent-style interfaces such as ReAct [5] suggest that LLMs benefit from interleaving reasoning steps with structured actions. We adopt this paradigm in the SAT domain via a structured scratchpad that enables models to store and retrieve learned clauses during inference.

2 Project Plan

- **Dataset and Traces:** We generate 100k 3-SAT formulas with 5–15 variables and trace their resolution using a static CDCL solver. A separate set of 5k OOD formulas with 16–25 variables is used to test generalization.
- **Training:** A transformer model is trained via next-token prediction on

textual traces that represent solver actions such as unit propagation and clause learning.

- **Inference with Scratchpad:** During inference, the model interacts with a structured memory. It emits actions like `WRITE_LEARNED_CLAUSE` and `READ_LEARNED_CLAUSES` to simulate learned clause management. This is conceptually similar to tool usage in ReAct-style agents [5].
- **Evaluation:** We evaluate the model using next-token accuracy and full-sequence trace correctness. The latter determines whether the model can solve entire OOD instances by generating a logically sound sequence of reasoning steps.
- **Future Work:** Reinforcement learning (RL) will be explored to optimize trace efficiency. Inspired by Kurin et al. [2], we aim to use RL to fine-tune decision-making and clause learning policies, possibly incorporating solver heuristics such as VSIDS.

3 Preliminary Results

In this phase, we limited training to the *analyze conflict* subroutine. Without any hyperparameter tuning, the model achieves over 95% next-token accuracy and nearly perfect trace reproduction on in-distribution examples. On OOD instances (16–25 variables), the model solves approximately 35% of cases end-to-end.

These results support the feasibility of training LLMs to emulate symbolic solvers through sequence prediction and structured memory interaction and this should enable easier analysis of these models than in the case of general-purpose reasoning where the solver is not known. Future work will include RL fine-tuning and integration of additional solver mechanisms to further improve generalization and efficiency.

References

[1] Borja Ibarz, Vitaly Kurin, George Papamakarios, Kyriacos Nikiforou, Mehdi Bennani, Róbert Csordás, Andrew Joseph Dudzik, Matko Bošnjak, Alex Vitvitskyi, Yulia Rubanova, Andreea Deac, Beatrice Bevilacqua, Yaroslav Ganin, Charles Blundell, and Petar Veličković. A generalist neural algorithmic learner. In Bastian Rieck and Razvan Pascanu, editors, *Proceedings of the First Learning on Graphs Conference*, volume 198 of *Proceedings of Machine Learning Research*, pages 2:1–2:23. PMLR, 09–12 Dec 2022.

[2] Vitaly Kurin, Saad Godil, Shimon Whiteson, and Bryan Catanzaro. Can q-learning with graph networks learn a generalizable branching heuristic for a sat solver?, 2020.

[3] David Mojžíšek, Jan Hůla, Ziwei Li, Ziyu Zhou, and Mikoláš Janota. Neural approaches to sat solving: Design choices and interpretability. *arXiv preprint arXiv:2504.01173*, 2025.

[4] Daniel Selsam, Matthew Lamm, Benedikt Bünz, Percy Liang, Leonardo de Moura, and David L. Dill. Learning a SAT solver from single-bit supervision. In *International Conference on Learning Representations*, 2019.

[5] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 10 2022.

[6] Xi Ye, Qiaochu Chen, Isil Dillig, and Greg Durrett. Satlm: satisfiability-aided language models using declarative prompting. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.