Sketching: a Cognitively inspired Compositional Theorem Prover that Learns

Brando Miranda[1]

Vision & motivation: Theorem proving has important applications in hardware and software verification. In hardware verification, it has been used for integrated circuit design [7, 8]. For software verification, a major success with theorem provers has been with the creation of CertC; a verified compiler for C; [9]. In the past, companies like Intel have made major investments in formal methods to guarantee the processors they manufacture do not have major floating-point bugs -- an important example of this is the Pentium FDIV bug in 1994 that cost them $500M [11]. As a consequence, theorem proving has been used to verify floating-point firmware [10]. But to effectively use this tool one requires highly trained human experts in the corresponding theorem prover and its area of application (e.g. using the HOL Light theorem prover for hardware verification). However, learnable automated theorem proving promises to revolutionize hardware and software verification by: 1) increasing the automatization of the challenging task of theorem proving and 2) by increasing the adaptability of such methods and therefore widening their use and applicability through learning. With this I envision that Nvidia can be a leader in this field by providing high quality verified GPU hardware, safe compilation of CUDA code, verified CUDA kernels, and exemplary verified interfaces of Nvidia GPU code for higher-level languages like python. With these vast immediately impactful applications I can see Nvidia leading the way to the highest quality products and saving millions of dollars in a fiercely competitive industry with increasing demands from gaming and deep learning applications.

In addition, theorem proving has had a major role in human history with the development of the scientific method, mathematics and technology. Thus, the last major part of my vision is to further empower humanity with powerful automation tools that can take all those areas even further. In addition, with a system that is capable of learning its applications are nearly endless. For example, it could play the crucial role of guaranteeing safe AI. Nvidia can become one of the first to pave the way forward in this exciting field and have a long-lasting impact for a richer, safer future.

Introduction: Current advances in Deep Learning have accelerated Artificial Intelligence (AI) in an unprecedented way especially in the domains of perception, machine translation, speech recognition, art generation, etc. However, most of the success has been on tasks that require primarily “system one” cognitive abilities [12] for example: intuitive, fast, unconscious, non-linguistic, habitual, or perception tasks. But intelligence also encompasses “system two” cognitive abilities [12], for example slow, logical, conscious, linguistic, algorithmic, reasoning, and mathematical abilities. In reality, intelligence encompasses both systems and one of the hallmarks of human intelligence is learning how to use both systems simultaneously to solve complex tasks. For this, I suggest we study the important task of theorem proving where the ability to employ both systems is crucial for the success of the system. I propose to tackle this problem by leveraging the insights provided by cognitive science through a technique called sketching [2]. Sketching is the ability to learn the higher level outline of a mathematical proof while delegating the smaller details to more specialized (potentially learned) reasoning or search methods. Sketching is powerful because it avoids the combinatorial explosion of proof search by leveraging the power of the idea of “divide and conquer” through sketching. In addition, my proposal is a novel conceptualization of theorem proving and explicitly explores the ability of an AI agent to learn how to effectively use system one and two cognitive abilities [12] together to solve an important task.

[1] *Department of Computer Science, University of Illinois at Urbana-Champaign.*
Brando Miranda <miranda9@illinois.edu>

Background: Sketching implements the cognitive science principle of compositionality for building programs by composing high-level concepts and then completing the details with a synthesizer. It's powerful because it avoids low-level reasoning and allows the synthesizer to avoid a combinatorial explosion during its search and completion of the sketch.

Research Question: Do cognitively inspired methods using sketching, combined with artificial neural networks, improve state performance on the modern interactive theorem proving (ITP) benchmark HOList [1]?

Research Plan & Methods: I propose working under the sketching framework suggested by Nye et at. [2] and extend it to interactive theorem proving. My extension will need the following: 1) a way to construct a formal proof sketch 2) an interactive theorem prover to build a synthesizer to complete and verify the proof and 3) a way to jump-start the process by processing the theorem to be proved. For the first part, I propose using Miz3 [3], a declarative proof language to construct formal proof sketches together with a sketcher Recurrent Neural Network (RNN) [2]. For the second part, I plan to use the theorem prover used by HOList [1] to build the synthesizer and verify completed proofs. For the third part, I propose using Graph Neural Networks (GNNs) [4] to compute a vector from the target theorem to initialize my whole proving system. The system will work as follows: first, I will compute a vector from the target theorem using the GNN and use it to initialize the sketcher RNN that will produce a formal proof sketch (using the Miz3 specification). Given a formal proof sketch, I will complete the proof using a trainable Monte Carlo Tree Search (MTCS) or using HOLight's hammer system [6]. A hammer is a technique to aid the prover interacting with the ITP by calling classical provers like Z3 [5] or HOL(y)Hammer [6] until one succeeds. After the proof is complete, the system will be trained to maximize the probability of proof completion under a distribution of time thresholds using gradient ascent as specified by Nye et al. [2] using the HOList proof data set [1]. Crucially, the verification of the proof will be done with HOList, thus allowing me to compare my progress with previous work [1,4].

Expected Results: Sketching is a method that leverages the power of compositionality for building proofs in an ITP. The expectation is that the sketcher will be able to break down the proof search, similar to how a divide-and-conquer method would, by outlining (i.e. sketching) the proof search and therefore allowing the search to be more efficient. Under a fixed time constraint, completing proofs more quickly also allows the system a larger throughput of successful proofs. Furthermore, it has another performance advantage that other systems lack: parallelism. This is because the details of a proof sketch can be completed in parallel because they have independent "holes" [2] to be filled.

Finally, the largest contribution to this work is the novel conceptualization of the process of theorem proving by examining ideas from cognitive science: in particular sketching, as a way to implement compositionality. My hope is that the success of this approach inspires AI researchers to expand to new horizons how we think about learning to reason and therefore open a novel and rich area of research in Artificial Intelligence.

Career Goals: My main career goal is to be an influential scientist in the field of Machine Learning and AI. I wish to contribute ideas that make us conceptualize AI in novel ways that end up being useful in the development of powerful - yet safe - Artificial General Intelligence (AGI). My preferred way to reach this goal is through a professorship where I can have independence, be a research leader, and mentor scientists of future generations. However, beyond any specific job - I want to engage in impactful research in academia, industry, or my own company. It's hard to predict now which one is the best but my ambition and commitment to this path is unwavering.

References:

[1] K. Bansal, S. M. Loos, M. N. Rabe; C. Szegedy and Wilcox, S. 2019. Holist: An environment for machine learning of higher-order theorem proving. ICML 2019. International Conference on Machine Learning.

[2] M. Nye, L. Hewitt, J. Tenenbaum, and A. Solar-Lezama. Learning to infer program sketches. arXiv preprint arXiv:1902.06349, 2019. [3] F. Wiedijk. A Synthesis of the Procedural and Declarative Styles of Interactive Theorem Proving. Logical Methods in Computer Science, 8(1), 2012. doi:10.2168/LMCS-8(1:30)2012.

[4] A. Paliwal, S. M. Loos, M. N. Rabe, K. Bansal, and C. Szegedy. Graph representations for higher-order logic and theorem proving. CoRR, abs/1905.10006, 2019.

[5] L. Mendonça de Moura and N. Bjørner. 2008. Z3: An Efficient SMT Solver. In Proceedings of the 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'08). Springer, 337–340.

[6] HOL(y)Hammer: Online ATP Service for HOL Light. C. Kaliszyk and J. Urban.

[7] Khan, W., Kamran, M., Naqvi, S. R., Khan, F. A., Alghamdi, A. S., & Alsolami, E. (2020). Formal Verification of Hardware Components in Critical Systems. *Wireless Communications and Mobile Computing*, *2020*. https://doi.org/10.1155/2020/7346763

[8] Li, L., Szygenda, S., & Thornton, M. (2005). Combining Simulation and Formal Verification for Integrated Circuit Design Validation.

[9] Berghofer, Stefan Strecker, Martin (2004). Extracting a formally verified, fully executable compiler from a proof assistant.

[10] Harrison, J. (2012). Formal Methods at Intel-An Overview.

[11] Harrison, J. (2003). Intel's Successes with Formal Methods.

[12] Bengio, Y. (2019). From System 1 Deep Learning to System 2 Deep Learning.