Electronic Circuit Optimization With Graph and Sheaf Neural Networks

Eva Hita Sogorb [1] **Cristian Sestito** [1] **Alexandros Keros** [1] **Themis Prodromakis** [1]

[1]*APRIL AI Hub, University of Edinburgh, Edinburgh, United Kingdom*. Correspondence to: Eva Hita Sogorb E.Hita-Sogorb@sms.ed.ac.uk.

1. Introduction

Moore's law refers to the observation that the number of transistors in an integrated circuit doubles every two years [1]. Since 1965, the growth in integrated circuit complexity had accurately followed this trend, until it started to plateau due to economical and physical barriers. Nowadays, the main physical constraints affecting system performance are related to power, heat generation, and the physics of complementary metal-oxide semiconductor (CMOS) transistor switching, all of which act as limiting factors [2]. In terms of economic limitations, Gordon Moore also observed a second law that described how the exponential increase in fabrication costs would become a limiting factor before physics [3].

As Moore's law becomes less applicable to the current technological development situation, new paradigms are needed to ensure power, performance and area savings (PPA) [4] for maintaining modern circuit-scaling demands. Approximate computing arises as a possible answer to this challenge, exploiting the trade-off between computational accuracy, power, and area requirements [5]. Gate-level pruning (GLP) stems from this paradigm by omitting gates from a circuit design while striving to maintain task-specific accuracy requirements. Current examples of tools that perform GLP require a time-consuming iterative simulation process. For example, a previous study [6] presented a probabilistic pruning-based tool [7] that is integrated in the digital circuit design flow to perform automatic GLP. This tool requires gate-level hardware simulations to be run in order to obtain a Switching Activity Interchange Format (SAIF) file to inform the decision of which wires are to be pruned. However, a more efficient approach that allows to skip the iterative simulation step is needed.

Artificial intelligence has been successfully applied across several domains to automate complex tasks by making accurate predictions, leading to significant time savings in fields such as healthcare [8]. Graph Neural Networks (GNN) provide the framework for performing learning and inference on combinatorial data, with successes in antibiotic discovery [9], where graphs encode molecular networks and their interactions, and in analogue circuit representation [10]. Sheaf Neural Networks (SNNs) appear as generalizations of GNNs, where information on vertices and edges is modeled as data in appropriate vector spaces, with consistency maps between them, forming what are known as cellular sheaves [11] [12]. SNNs have been successfully used in applications such as recommendation systems, outperforming GNNs [13].

GNNs have been explored in EDA problems such as congestion prediction [14], testability analysis [15], and classification of sub-circuits in netlists for reverse engineering (RE) purposes. For example, DeepGate is a GNN-based solution for the representation learning of logic gates [16]. The purpose of this model, as well as that of its subsequent versions such as DeepGate2 [17], is to create a general-purpose neural representation framework for logic circuits, as well as a circuit-aware GNN architecture that can be applied as the basis to solve several EDA problems. However, even the latest version of the algorithm, DeepGate4, is susceptible to the inherent limitations presented by GNNs, such as oversmoothing, which suggests that another approach might be necessary.

We hereby show the representation of digital circuits as sheaves (Figure 1), and propose the use of SNN to learn and automate GLP. By representing operational characteristics, power demands, and area requirements concisely as stalks, and circuitry operations as restriction maps of a *circuit sheaf*, we propose to learn a sheaf diffusion operator that performs GLP.

2. Methods

This section introduces the concept of GLP and provides an overview of SNN. It then describes the project framework, from the creation of a dataset to model training and evaluation, highlighting the integration of sheaf neural networks and GLP.

2.1 Gate-Level Pruning

GLP is a technique that can be used to trade off accuracy for PPA savings by removing the parts of the circuit with the lowest contribution towards accuracy. There are several parameters that need to be considered when performing GLP. Firstly, the error tolerance, which depends on the application of the circuit to be pruned. Checking that the error introduced by pruning is below the threshold set by the designer is the first step in a GLP pipeline. Secondly, the significance and activity or toggle count (TC), which are the two criteria that can be used to decide which nodes should be pruned. The significance-activity product (SAP) indicates which nodes are pruned first, starting with those with a lower SAP, while the TC measures how many times a signal changes state.

2.2 Sheaf Neural Networks

A cellular sheaf is a combinatorial structure generalizing graphs that represents data on vertices and

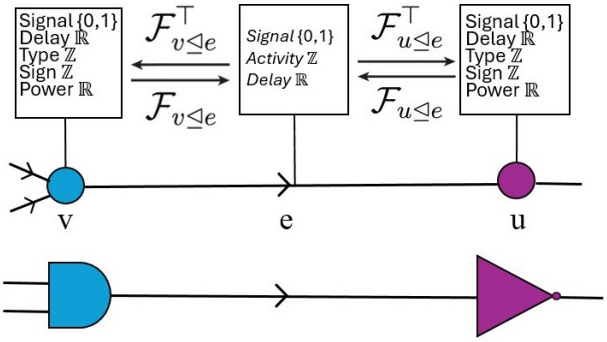

Fig. 1: An example showing a simple digital circuit and its corresponding sheaf representation. The vector spaces (represented as squares) contain information relevant to each of the edges and nodes of the sheaf. The restriction maps (represented by $\mathcal{F}^{\top}$) correspond to the learnable part of the SNN.

edges as sheaves i.e. elements of some vector space, with consistency maps, called *restriction maps*, between them [12]. While graphs can only represent the structure of the data being used, sheaves provide a way to capture how information at different parts of a graph is related. In the context of digital circuits, graphs represent logic gate connectivity (Figure 1), whereas sheaves can store information on the relationship between the input and the output of a logic gate. In the application discussed in this paper, sheaves provide a way to store SAP information for each of the edges, and to capture the complex relationship between different circuit components, thus informing the SNN-led GLP process.

Sheaf neural networks assume that restriction maps are parametrized and aim to learn them through *sheaf diffusion* [18]. SNNs address the two main limitations arising in GNNs: heterophily and oversmoothing. The heterophily problem describes the underperformance of GNNs on graphs where neighboring nodes do not have similar features, as they are designed to assume homophily or similarity between neighboring nodes. Oversmoothing refers to the reduction in performance as the depth of a GNN model is increased. As such, SNNs are ideal candidates for learning to automate GLP, as they are able to consistently represent diverse node data while not being hindered by the limitations that plague GNNs.

2.3 The Framework

Firstly, we are generating a dataset of full-precision and approximate arithmetic circuits. The AMD Vivado tool is being used to obtain the switching activity of the wires in these circuits through simulations. Then, circuits are being represented as sheaves by equipping them with the data extracted from Vivado, with information such as activity being attached as stalks to edges on the sheaves. These will then be used to train a SNN to automate GLP by learning the restriction maps relating vectors and edges. This is a classification problem where the SNN will learn to classify nodes and edges as either prune or keep. A multi-part loss function will be used during training, as in Equation 1:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{area}} + \mathcal{L}_{\substack{\text{toggle} \\ \text{count}}} + \mathcal{L}_{\text{accuracy}} + \mathcal{L}_{\text{power}} \quad (1)$$

The total loss consists of several components: the area loss, $\mathcal{L}_{\text{area}}$, which accounts for how much area the circuit uses; the toggle count loss, $\mathcal{L}_{\text{toggle count}}$, which measures how many times the gates change state; the accuracy loss, $\mathcal{L}_{\text{accuracy}}$, which measures how the accuracy changes depending on the accuracy of the circuit; and the power loss, $\mathcal{L}_{\text{power}}$, which accounts for the energy consumption of the circuit.

We will eventually evaluate the effectiveness of the learned pruning by comparing the approximate circuits (post-pruning) with the full-precision counterparts (before pruning).

Acknowledgments

We acknowledge support of the UKRI AI programme, and the Engineering and Physical Sciences Research Council, for APRIL- EPSRC AI Hub for Productive Research & Innovation in electronics [grant number EP/Y029763/1]. The 2025 Research placements were supported by Google DeepMind, the Hg Foundation and the Royal Academy of Engineering under the Google DeepMind Research Ready scheme [grant number GDMRR-2425-1-156].

References

[1] Gordon E. Moore. Cramming more components onto integrated circuits. *Electronics*, 38(8):114–117, April 1965. Reprinted in *Proceedings of the IEEE*, Jan 1998.

[2] Theis, Thomas N. et al. The end of moore's law: A new beginning for information technology. *Computing in Science & Engineering*, 19(2):41–50, 2017.

[3] R.R. Schaller. Moore's law: past, present and future. *IEEE Spectrum*, 34(6):52–59, 1997.

[4] Parasyris, Konstantinos et al. Approximate computing through the lens of uncertainty quantification. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–14, 2022.

[5] Menon, Harshitha et al. Approximate high-performance computing: A fast and energy-efficient computing paradigm in the post-moore era. *IT Professional*, 25(2):7–15, 2023.

[6] Schlachter, Jeremy et al. Design and applications of approximate circuits by gate-level pruning. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 25(5):1694–1702, 2017.

[7] Lingamneni, Avinash et al. Energy parsimonious circuit design through probabilistic pruning. In *2011 Design, Automation & Test in Europe*, pages 1–6, 2011.

[8] Bajwa, Junaid et al. Artificial intelligence in healthcare: transforming the practice of medicine. *Future Healthcare Journal*, 8(2):e188–e194, Jul 2021.

[9] Stokes, Jonathan M. et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702.e13, 2020.

[10] Sirico, Anthony, Jr. et al. On the use of geometric deep learning for the iterative classification and down-selection of analog electric circuits. *Journal of Mechanical Design*, 146(5):051703, 11 2023.

[11] Jakob Hansen and Thomas Gebhart. Sheaf neural networks. *CoRR*, abs/2012.06333, 2020.

[12] Barbero, Federico. Attention-based sheaf neural networks. M.phil. dissertation, University of Cambridge, Department of Engineering, August 2022. Submitted August 2022.

[13] Purificato, Antonio et al. Sheaf4rec: Sheaf neural networks for graph-based recommender systems, 2024.

[14] Robert Kirby, Saad Godil, Rajarshi Roy, and Bryan Catanzaro. Congestionnet: Routing congestion prediction using deep graph neural networks. In *2019 IFIP/IEEE 27th International Conference on Very Large Scale Integration (VLSI-SoC)*, pages 217–222, 2019.

[15] Yuzhe Ma, Haoxing Ren, Brucek Khailany, Harbinder Sikka, Lijuan Luo, Karthikeyan Natarajan, and Bei Yu. High performance graph convolutional networks with applications in testability analysis. In *Proceedings of the 56th Annual Design Automation Conference 2019*, DAC '19, New York, NY, USA, 2019. Association for Computing Machinery.

[16] Min Li, Sadaf Khan, Zhengyuan Shi, Naixing Wang, Huang Yu, and Qiang Xu. Deepgate: learning neural representations of logic gates. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, DAC '22, page 667–672, New York, NY, USA, 2022. Association for Computing Machinery.

[17] Zhengyuan Shi, Hongyang Pan, Sadaf Khan, Min Li, Yi Liu, Junhua Huang, Hui-Ling Zhen, Mingxuan Yuan, Zhufei Chu, and Qiang Xu. Deepgate2: Functionality-aware circuit representation learning. In *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, pages 1–9, 2023.

[18] Barbero, Federico et al. Sheaf neural networks with connection laplacians, 2022.