Photonic Interconnect Based Neural Network Simulator

Andy Wolff, Kyle Shiflett, and Avinash Karanth
School of Electrical Engineering and Computer Science, Ohio University
Email: {aw415517, ks117713, karanth}@ohio.edu

***Abstract*—This paper introduces a photonic interconnect simulator that supports architecture simulation and design on a large enough scale to facilitate design-space exploration of optical neural networks on a C++ implementation with PyTorch integration.**

I. Introduction

Photonic hardware accelerators offer advantages in bandwidth, and power consumption for implementation of large-scale computing systems [1]. This property makes them attractive for implementation of architectures such as neural networks which benefit from increased parallelism and bandwidth. A significant challenge to designing photonic systems is a lack of tools oriented towards designing scalable photonic interconnect based architectures. There certainly exist tools that are capable of simulating photonics [2], but a tool specifically designed with the scalability and the abstraction needed to design and explore a system quickly and accurately has yet to exist. In order to perform a full design-space exploration, a tool for photonic simulation must be simple enough to allow for rapid creation and adjustment of test architectures, and allow for encapsulation of portions of a system in order to simplify larger, system scale design choices and simulations. The photonic network simulator must implement different photonic devices and circits while retaining accuracy in signals, losses, latency, and power consumption. In this paper, we introduce a simulator built in C++ that is capable of simulating photonic components, and enables users to create layers of abstraction in order to simulate larger and more complicated systems easily without sacrificing accuracy.

II. Simulator Design

The most important part of the simulator is the optical signal implementation. Optical signals are represented as a complex PyTorch tensor whose elements correspond to a particular wavelength. Signals are stored as an array of independent, non-interacting wavelengths of light. A combination of different wavelengths in a signal can allow for complicated signal patterns when examining total power over time. By using complex numbers, a single element can provide both phase and magnitude information for a particular wavelength. The set of wavelengths represented within the simulation is readily configurable and modifications to wavelength parameters take effect for the entire system. Propagation of signals over time is handled by running the simulation as a sequence of time steps, with both the duration of the time step and the total time for the simulation as configurable. Each time step every component takes the inputs at each of their ports and produces the corresponding output.

Components of the circuits can be modeled by separating them into discrete component blocks [4]. Components are represented as C++ child classes of a broader component parent class. This parent class, and the environment connected to it, automatically implements some basic component functionality, such as handling for the number of ports, managing of links to other components, and automatically adding the component to the simulation environment on creation. Placing components in container classes allows most simple pieces of an optical design to be created in single line of code. Many simple components, such as the Y-branch can be represented with a matrix multiplication [3]. These models take input signals as a vector and operate on both phase and magnitude through matrix multiplication.

All linear algebra performed in the simulator is implemented within PyTorch. This allows utilization of the GPU for tensor manipulation, accelerating the simulator significantly. Combining PyTorch with conventional multithreading allows for as high a utilization of hardware as possible, decreasing the time required for simulation. Additionally, if all of the simulated wavelengths operate independently, as is typically the case for optical simulation, the simulation can be divided amongst multiple machines with the same simulation programmed, but a different portion of the wavelength set.

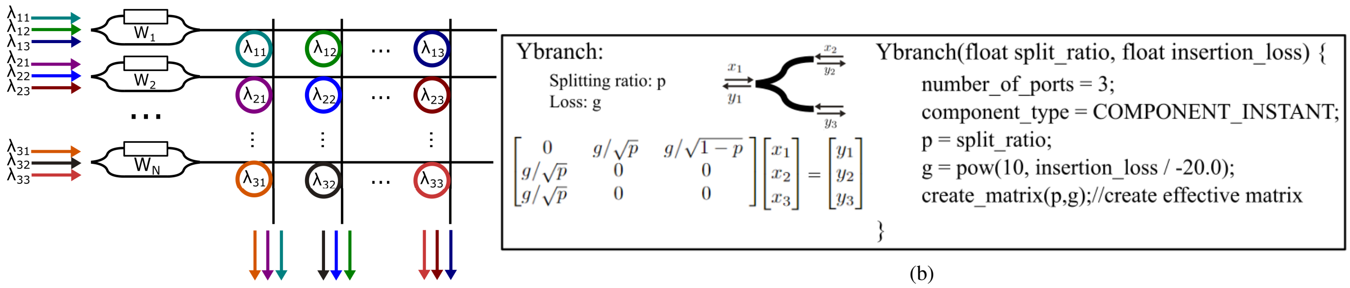
III. Applications

The task of combining preexisting components to form new ones is straightforward. With optical couplers and waveguides, one may implement a ring resonator. The resulting resonator is then wrapped in a new resonator class. This resonator class can map the linking functions so that this new composite component behaves in simulation construction as any lone component does. Many of these resonators are then linked together to form a grid, and combined with another class implementing a Mach-Zehnder interferometer in order to implement a single multiply and accumulate operation as in Figure 1. This is moved a step further by wrapping the multiply and accumulate implementation into yet another container. This new implementation

This research was partially supported by NSF grants CCF-1513606, CCF-1703013 and CCF-1901192.

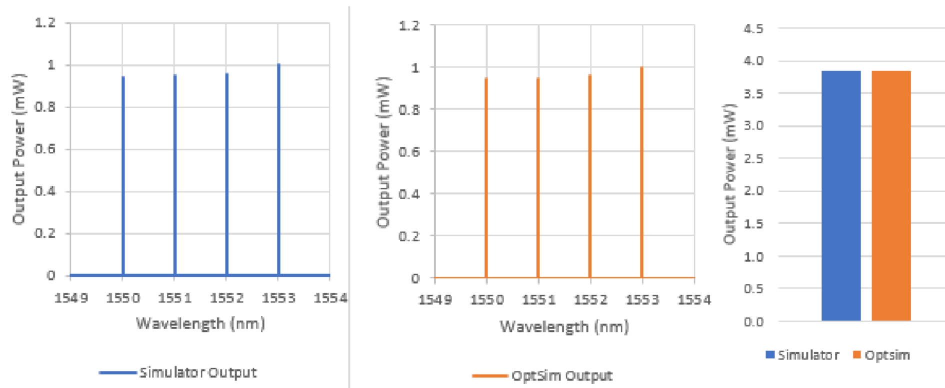
978-1-6654-8655-2/23/$31.00 ©2023 IEEE

2023 IEEE Silicon Photonics Conference (SiPhotonics) | 978-1-6654-8655-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/SIPHOTONICS55903.2023.10141962



(a) (b)

Fig. 1. **(a) Optical dot product circuit (b) Matrix multiplication implementation of a Y-branch component, and pseudocode for Y-branch constructor**



(a) (b)

Fig. 2. **(a) Simulated power frequency diagram for optical dot product, using 4 channels (b) Total simulated output power. The result of the optical dot product**

has an added support in that the number of inputs for a multiply and accumulate is variable, and the class is constructed so that the number of input and output channels can be set at construction. This allows for creation of many different versions of this implementation quickly, allowing for determining its efficiency and testing its parameters much more quickly than if one attempted to construct many different implementations of this architecture by hand. Critically, the process of creating new combined components does not have to change as more and more levels of abstraction are added. At a system level, this allows users to place subsystems into black-box components after they are implemented once, allowing for larger and larger systems without making the simulator more unwieldy.

The optical dot product implementation was placed within the simulator and the results compared to that of another simulator, Optsim, with the same system as in Figure 2. In both cases, the distribution of power per wavelength, and then the total optical power for all wavelengths was taken in Watts. In each case there is some loss due to the way resonators function for every resonator that a signal passes. The power distribution is nearly identical. Numerically, the only significant difference in simulation results was that the resolution of wavelengths was slightly different, giving a slight change in the exact wavelength with the peak power.The final results for the optical dot product are identical up to 4 significant figures. The small difference in the exact value is likely due to compounded small rounding operations due to the limitations of floating point precision.

IV. Conclusions

This paper introduced a scalable photonic hardware simulator in order to create a photonic simulator compatible with larger photonic neural network designs. An optical matrix multiplication was implemented, the power overhead was computed, and the results of the simulation were compared to the results of similar optical simulator, and found to be comparable.

References

[1] E. M. et al., "Optical interconnect system integration for ultra-short reach applications," *Intel Technology J.*, vol. 8, no. 2, pp. 115–128, May 2004.
[2] F. Laporte, J. Dambre, and P. Bienstman, "Highly parallel simulation and optimization of photonic circuits in time and frequency domain based on the deep-learning framework PyTorch," *Sci. Rep.*, vol. 9, no. 1, p. 5918, Apr. 2019.
[3] Z. Lin and W. Shi, "Broadband, low-loss silicon photonic y-junction with an arbitrary power splitting ratio," *Opt. Express*, vol. 27, no. 10, pp. 14 338–14 343, May 2019. [Online]. Available: http://www.opticsexpress.org/abstract.cfm?URI=oe-27-10-14338
[4] D. Melati, F. Morichetti, A. Canciamilla, D. Roncelli, F. M. Soares, A. Bakker, and A. Melloni, "Validation of the building-block-based approach for the design of photonic integrated circuits," *Journal of Lightwave Technology*, vol. 30, no. 23, pp. 3610–3616, 2012.