Optically Connected and Reconfigurable GPU Architecture for Optimized Peer-to-Peer Access

Erik Anderson
Columbia University
efa2113@columbia.edu

Jorge González
University of Campinas
jorge.gonzalez@ic.unicamp.br

Alexander Gazman
Columbia University
ag3529@columbia.edu

Rodolfo Azevedo
University of Campinas
rodolfo@ic.unicamp.br

Keren Bergman
Columbia University
kb2028@columbia.edu

ABSTRACT

Increasing industry interest in the optimization of inter-GPU communication has motivated this work to explore new ways to enable peer-to-peer access. Specifically, this paper investigates how reconfigurable optical links between GPUs in multi-GPU servers can allow for minimized memory transfer latencies for given machine learning applications. Silicon photonics (SiP) is proposed as the enabling technology for such a reconfigurable architecture due to the potential for scalable and cost-efficient production. We evaluated our architecture using traffic obtained from an NVLink-connected 8 GPU server executing a set of machine learning models including AlexNet, DenseNet, NASNet, ResNet, MobileNet, and VGG16. Our results show up to 24.91% reduction of the total relative transmission latency (RTL) between peers.

CCS CONCEPTS

• **Computing methodologies** → *Graphics processors*; • **Hardware** → *Emerging optical and photonic technologies*;

1 INTRODUCTION

Since the introduction of GPUDirect peer-to-peer access [1], NVIDIA has been gradually placing more and more importance on the efficient movement of data between device memories. While providing substantial performance benefits, adding multiple GPUs to a system necessitates non-local GPU memory accesses. Because of this, implementing efficient memory copies between devices has become essential for executing modern GPU applications, such as those used for various machine learning models.

NVLink was first introduced in 2014 [2] to replace the severely bandwidth-limited PCIe connections that were initially used to provide peer-to-peer access. NVIDIA's 2018 release of NVSwitch [2] further highlights the importance of enabling efficient interconnectivity between peers in GPU-accelerated servers. In this paper, an optically connected GPU architecture using Silicon Photonic (SiP) circuit switches is proposed as a radically new GPU communication structure to further increase performance. An exhaustive bandwidth-steering algorithm is introduced to minimize total latency for device memory transfers. Initial results for this algorithm and proposed architecture are reported using traffic traces generated from a multi-GPU system.

MEMSYS, October 1–4, 2018, Old Town Alexandria, VA, USA

ACM ISBN 978-1-4503-6475-1/18/10.
https://doi.org/10.1145/3240302.3240418

2 PHOTONIC GPU ARCHITECTURE

2.1 Reconfigurable Silicon Photonic Links

Through leveraging the immense investment already placed in the microelectronics industry, Silicon Photonics (SiP) has enabled cost-efficient production of high-speed on-chip optics for interconnecting and disaggregating computing systems [6].

In addition, SiP optical circuit switches (OCS) allow dynamic link-level reconfigurability for a new class of bandwidth-steered applications. In [7], it was shown that bandwidth-steering via OCS can achieve significant performance gains for dragonfly networks. Using SiP, optical links can be traded between groups to allow for optimized inter-group connectivity.

Fig. 1 shows a simple example of this bandwidth-steering concept to optimize the outgoing connections from 1 GPU in a 3 GPU system. Depending on the microarchitecture of the OCS, the optical signals in Fig. 1 could each represent a single wavelength signal *or* a wavelength-division multiplexed (WDM) signal. The OCS is configured at the beginning of each application phase to provide larger portions of the total bandwidth to the highest-traffic GPU pairs. Fig. 1b illustrates a situation in which the algorithm has determined that an unequal amount of bandwidth between GPU A and GPUs B and C will result in increased performance. The specifics of this configuration algorithm are detailed in the following section.

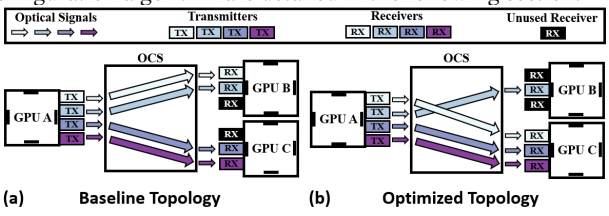
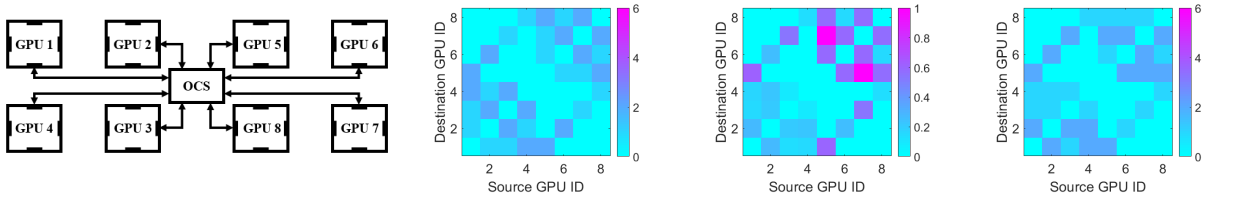


Figure 1: Based on the relative amounts of data sent from GPU A to GPUs B and C during a given application, the optical circuit switch (OCS) will reconfigure itself to minimize GPU A's total transmission latency. (a) The baseline configuration provides equal bandwidth from GPU A to GPUs B and C (b) In this example, the optimization algorithm has determined that GPU A requires 3x more bandwidth to GPU C than GPU B.



(a) 8 GPU configuration connected using an optical circuit switch (OCS) **(b) Original Hyper-Mesh Topology** **(c) Traffic Matrix - Flowers Dataset - MobileNet model** **(d) Optimized Connectivity Matrix**

Figure 2: Optimizing connectivity for 8-GPU server using Flowers training set on a replicated-variable MobileNet model.

2.2 Link Optimization Algorithm

For the state-of-the-art NVLink topology, much of the complexity of the bandwidth-steering approach from [7] can be reduced to a minimization problem. Because NVLink does not support routing of any kind, all existing flows between GPUs must be supported by some non-zero amount of bandwidth. After reserving the minimum amount of allocatable bandwidth, termed a *bandwidth unit*, for every required connection, the remaining bandwidth units are distributed among the other GPUs to minimize the total amount of relative transmission latency (RTL). To calculate the RTL for a single source-destination pair requires the division of the element `[i][j]` in the traffic matrix for a given application by the corresponding element in the topology's connectivity matrix, where `[i]` and `[j]` represent the destination and source GPU IDs respectively. An aggregate RTL for each GPU is calculated by summing the RTLs from the appropriate column in the traffic and connectivity matrices. An exhaustive algorithm is used to iteratively minimize the aggregate RTL for each GPU. This requires calculation of $c_j * \binom{k-1}{k-c_j}$ RTLs per GPU where c_j is the number of required connections for GPU j, and k is the number of bandwidth units per GPU.

By analyzing each GPU independently, the algorithm generates a new topology without considering the maximum number of available receivers per GPU. However, the cost of additional receivers to accommodate this approach can be minimized by optically disaggregating the GPUs to allow for extra physical space per device.

3 EXPERIMENTAL RESULTS

Fig. 2a shows our proposed reconfigurable architecture for 8 GPUs within a single server. The topology is based on the NVLink-connected hyper-mesh GPU configuration. The connectivity matrix for this topology is shown in Fig. 2b. Each element `[i][j]` in the connectivity matrix denotes the total number of bandwidth units that GPU `j` can use to send data to GPU `i`. In this experiment, each bandwidth unit represents 1 NVLink sublink that provides 25 GBps of bandwidth. Each NVLink-enabled GPU has 6 outgoing and incoming sublinks, giving a total of 300 GBps of bidirectional bandwidth [2]. It is important to note that while Fig. 2a shows the outgoing and incoming connections as a single bidirectional link, the optimization algorithm treats each unidirectional connection independently.

We evaluated our architecture using the training phase of several Tensorflow machine learning models with both parameter server and replicated variables [4]. The Cifar10 dataset [5] was used to train the AlexNet, DenseNet100, DenseNet40, NASNet, ResNet110, and ResNet20 models. The Flowers dataset [3] was also used, and provided input to both MobileNet and VGG16 models.

A geometric average of 10% and 4% reduction in total RTL is achieved when optimizing connectivity using replicated and parameter server variables respectively. The maximum percentage decrease in RTL using replicated variables was 24.77% using the Flowers training set on the AlexNet model, and the minimum was 1.09% for ResNet110 with Cifar10 input. For parameter server variables, the maximum was 24.91% for AlexNet with Cifar10 input, and the minimum was 0.52% for NasNet with Cifar10 input. Only a small reduction is seen when the original hyper-mesh topology is well suited to the application. As expected, the replicated variable models experience increased performance relative to the parameter server models. This is due to higher levels of peer-to-peer traffic as a result of variable synchronization. The traffic matrix for the replicated MobileNet model with Flowers training set is shown in Fig. 2c. Each value in the traffic matrix is normalized by the maximum number of bytes sent between any two GPUs. Using the equation detailed in Section 2.2, only 320 RTL calculations are required to perform the optimization algorithm. The generated connectivity matrix is shown in Fig. 2d, and provides a 10.61% reduction in RTL at the cost of only 2 additional receivers per GPU.

4 CONCLUSION AND PROPOSED WORK

In conclusion, we have proposed a novel optically-connected GPU architecture that uses an exhaustive minimization algorithm for bandwidth-steering multi-GPU systems. Initial results show a significant decrease in total RTL at the cost of a few additional receivers per GPU. In future work, we will analyze the trade-offs between reduced latency and additional hardware, and determine how other applications, such as Big Data, could benefit from this architecture.

ACKNOWLEDGMENTS

This work was supported by the São Paulo Research Foundation (FAPESP-2014/016429), CAPES (PROCAD 2966/2014) and the NSF IGERT Program (DGE-1069240).

REFERENCES

[1] [n. d.]. NVIDIA GPUDirect. Retrieved April 24, 2018 from developer.nvidia.com/gpudirect
[2] [n. d.]. NVIDIA NVLink Fabric. Retrieved April 23, 2018 from www.nvidia.com/en-us/data-center/nvlink/
[3] [n. d.]. Tensorflow Flowers Dataset. Retrieved April 28, 2018 from download.tensorflow.org/example_images/flower_photos.tgz
[4] [n. d.]. Tensorflow High-Performance MOdels. Retrieved April 28, 2018 from www.tensorflow.org/performance/performance_models
[5] Alex Krizhevsky and Geoffrey Hinton. 2009. *Learning multiple layers of features from tiny images.* Technical Report. University of Toronto.
[6] C. Sun et al. 2015. Single-chip microprocessor that communicates directly using light. *Nature* 528, 7583 (2015), 534–538.
[7] K. Wen et al. 2017. Flexfly: Enabling a Reconfigurable Dragonfly through Silicon Photonics. *International Conference for High Performance Computing, Networking, Storage and Analysis, SC* November (2017), 166–177.