Latency-avoiding Dynamic Optical Circuit Prefetching Using Application-specific Predictors

Ke Wen[1], Sébastien Rumley[1], Jeremiah Wilke[2], and Keren Bergman[1]

[1]Depart. of Electrical Engineering, Columbia University, New York, NY 10027, USA
[2]Scalable Modeling and Analysis, Sandia National Labs, Livermore, CA 94551, USA

Abstract. Optical circuits are capable of providing large bandwidth improvements relative to electrical-switched networks in high-performance computing (HPC) systems. However, due to the circuit switching nature of optical systems, setup delays may prevent HPC systems from fully utilizing the available bandwidth. This paper proposes an application-guided circuit management technique that can achieve latency-avoiding dynamic reconfiguration, better leveraging the high-bandwidth photonics and accelerating system performance. By learning the temporal locality and communication patterns from upper-layer applications, the technique not only caches a set of circuits to maximize reuse, but also prefetches predicted circuits to actively hide the setup latency. We apply the technique to communication patterns from a spectrum of science and engineering applications. The results show that setup delays via circuit misses are significantly reduced, showing how the proposed technique can improve circuit switching in HPC optical interconnects.

Keywords: optical interconnection network, high-performance computing, circuit switching, cache, prefetching, latency-avoiding

1 Introduction

Optical communication systems can provide large bandwidth reaching the terabit/s order [1–4]. Furthermore, due to the low loss of optical media, optical signals remain almost unaffected after transmission over warehouse-scale distance, leading to extremely attractive energy efficiency. These capabilities makes optical communication a natural candidate for scaling the end-to-end data movement capability of large-scale high performance computing (HPC) systems [5–9].

Despite the above advantages, optical interconnects also have a set of special operation requirements. For example, resonance-based silicon photonic devices such as microring resonators require a wavelength tuning process to reach the designed operating wavelengths [10]. Also, these devices can be sensitive to the ambient temperature due to the high thermal-optic constant of silicon, thus may require thermal re-initialization during path setup in case the temperature has changed since last operation [11]. The above requirements can lead to longer link

initialization delays than is required by conventional electronic links. Furthermore, mature optical communication techniques often rely on circuit switching, which can induce additional path setup delays. The above delays can add to the execution time of HPC applications, preventing efficient use of the high optical bandwidth for application speedup. Thus, it is extremely important to develop techniques that can hide these delays from the upper-running application.

A potential method to hide the circuit setup latency, as proposed by [12], is to manage optical circuits in such a way that a communication request can immediately find the required circuit upon its arrival (we call this a *circuit hit*). One way to enable circuit hits is to explore the temporal communication locality in an application and allow a node to maintain circuits towards a set of frequently addressed destinations. If a circuit already connects to the required destination, messages can be immediately transmitted, eliminating the need for circuit setup. Furthermore, by reusing a circuit multiple times within its maintenance period, the scheme can effectively amortize the setup delay compared to a per-message setup scenario. Based on the above principle, we previously proposed a circuit-cached interconnection architecture that maintains a set of optical circuits towards recently frequently-accessed neighbors [12]. Such multi-circuit capability, from a device perspective, has been recently enabled by densely-integrated silicon photonic network interfaces [13–15] as well as the selectivity towards wavelength-division multiplexed (WDM) or mode-division multiplexed (MDM) channels [16–25]. The work [12] also utilizes cache-like techniques to design the circuit replacement policy in order to enhance the hit rate. This circuit-cached architecture, however, relies on circuit misses to update the cached circuit set – that is, the architecture does not replace a circuit until a miss occurs. This reactive strategy constitutes a drawback: if a required circuit is not yet in the cache set, its setup latency will never by hidden

To address the reactive replacement problem, this work further proposes an active circuit prefetching approach based on the circuit-cached scheme in [12]. By prefetching circuits before real requests arrive, the approach can further reduce or even eliminate the setup delays. Specifically, an application-specific predictor is proposed to learn characteristic *predecessor-follower* destination patterns in an application's communication behavior. Simulation based on a broad spectrum of benchmarks shows that the proposed caching-plus-prefetch scheme significantly enhance the hit rate performance compared to the previous caching-only scheme (by a rate increase as large as 95%).

2 Related Work

Optical interconnection networks have been vastly investigated as one of the next-generation high-bandwidth network solutions for high-performance computing [8, 26–29]. In addition, several works [5, 6, 31, 12] have specifically explored the use of optical circuits in the context of computing. Shalf et al [5] and Barker et al [6] propose the use of optical circuit switching with electrical packet switching in a hybrid network. However, neither of these two works considers

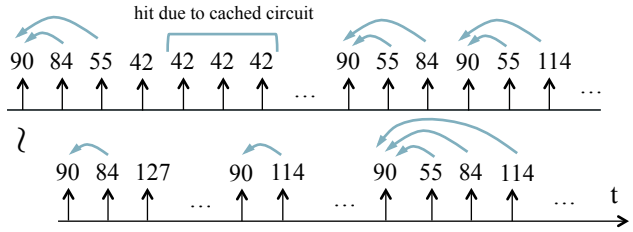

Fig. 1. A sequence of destinations that a rank of a parallel *Adaptive Mesh Refinement* application communicates to. The *blue* arrows indicates characteristic *predecessor* (**90**) - *follower* (**84**, **55**, **114**) patterns.

hiding circuit setup delays from the application. Hendry [31] considers avoiding the setup delay by inserting explicit circuit setup commands in the application code ahead of real communication calls. However, this method may increase the programmer's burden and entangle the circuit management with the real computation workload. In comparison, a recent work by Wen et al [12] abstracts the circuit management from the application, and proposes a circuit-cached architecture that maintains a set of established circuits to leverages reuse opportunities. By optimizing the replacement policy in a cache-like manner, the architecture is able to achieve high hit rates and amortize the setup delays. However, the architecture relies on passive replacements created by circuit misses to update the cached circuit set, resulting in limited performance.

To better avoid the setup latency, this paper builds on the circuit-cached architecture of [12] and augments it by adding an active prefetch capability. Although Hendry [31] also employs an active latency-avoiding approach, this work differs from it by 1) eliminating the need for inserting explicit circuit setup commands in the application and 2) amortizing the setup delay by allowing circuit caching and reuse.

3 Circuit Prefetch Using Application-specific Predictors

3.1 Caching-Plus-Prefetch

Accurate prediction of incoming circuit requests is critical to efficient circuit prefetching. The prediction can be made by learning the communication behavior of an upper-running application. In this paper, we consider the learning process in an online fashion and use as learning material the destination sequence generated by a network end point, for example, an application rank. In particular, we are interested in extracting characteristic *predecessor-follower* patterns from the destination sequence. These patterns, repeated due to workload iterations, can provide useful information regarding which circuit would be requested after the current one, thus enhancing the prediction accuracy.

Fig. 1, for example, shows a sequence of destinations addressed by one rank of a parallel *Adaptive Mesh Refinement* (AMR) application along the time axis. In

this example, circuit requests towards destinations **84**, **55** and **114** often follow the request towards destination **90**. With knowledge of such a *follower* pattern, a circuit management runtime can prefetch the most probable *follower* circuits when still communicating with the current destination, hiding the setup delay from the application. It should be noted that although a caching-only approach without prefetching can also achieve circuit hits when there are recurring communication requests – for example, the repeated requests to **42** in Fig. 1 – it misses the *follower* pattern that could further reduce the setup latency.

3.2 Prefetch Predictions

Predicting the follower circuits is an important step of effective prefetching. Several prediction techniques have been proposed in the realm of cache management [32–34]. These techniques, however, are designed for optimizing memory accesses, which often have constant access strides in the address space (e.g. when accessing an array in the memory). Hence, a prediction is often made by adding or subtracting a constant stride to or from the current address. The circuit communication considered in this paper, by contrast, does not always have a constant "stride" in the destination ID sequence. Such a lack-of-stride feature is even more common for applications with irregular communication patterns. Therefore, conventional stride-based cache prefetch techniques may not work for the circuit communication scenario considered in this paper.

To solve the above problem, the circuit prefetch runtime proposed in this work uses a *lookup table* (LUT) to learn the characteristic follower patterns. The LUT, maintained by each node, uses a *predecessor* ID as an entry's key, and the corresponding *followers'* ID (with their repeat frequency) as the entry's value. By observing the local communication history of an application rank, the node constantly updates the LUT to record the k most frequent *followers* of each *predecessor*. Here, we call parameter k the *tail length*, the maximum number of *follower* circuits for a given *predecessor* circuit. In Fig. 1, the predecessor **90** has three followers **55**, **84**, **144**, giving a tail length $k = 3$. The tail length impacts the actuation costs induced by prefetching, including the power consumed by circuit setup. The tail length also determines the size of the LUT and the update complexity. Thus, the tail length k imposes a trade-off effect in circuit prefetching: increasing k may result in more prefetched circuits and hence a potentially higher hit rate, but it may also induce more actuation energy consumption, a larger LUT footprint and higher update complexity. We will analyze such trade-offs in the next section.

When a circuit request arrives and is recognized as a recorded *predecessor*, its correspondent follower circuits will be prefetched by the circuit management runtime, unless the follower circuit is already cached. In this paper, we assume that the replacement policy used for prefetching is the same as that used for circuit misses. We also assume that only vacant circuits that are not in data transmission can be considered as a replacement candidate. Furthermore, upon circuit misses (hard replacement), a prefetched circuit can be preempted.

Table 1. Description of benchmarks used in the simulation and their communication features (numbers mearsured at 256 ranks).

Application	Description	Neighbor-to-Rank Ratio	Reuse Distance [12]
HPCCG [35]	Conjugate gradient code for 3D chimney domain simulation	4.69%	[8,16)
miniFE [35]	Unstructured implicit finite element codes	10.04%	0, [8,16)
miniMD [35, 36]	Molecular dynamics for spatial-decomposition particle simulations	5.18%	1, [8,16)
LULESH [37, 38]	Livermore Unstructured Lagrangian Explicit Shock Hydrodynamics	17.71%	[32,64)
Multigrid [39]	Differential equations solver using a hierarchy of discretizations	14.46%	[32,64)
CNS [39]	Compressible Navier Stokes equations with constant viscosity and thermal conductivity	17.30%	[32,64)

4 Performance Evaluation

4.1 Methodology

To evaluate the effectiveness of the proposed prefetch scheme, we compare its performance with the caching-only scheme in Ref. [12]. In both cases, a *Least Recently Used* (LRU) replacement policy is used. We use multiple mini-apps that cover a wide spectrum of scientific computations as upper-running benchmarks. These benchmarks, representing different communication patterns, are simulated based on a non-blocking circuit-switched network implementing one of the two latency-avoiding schemes above. The applications are simulated using trace replay from a library of DUMPI traces in conjunction with the macroscale components of the SST simulator [40]. Time gaps between consecutive communication calls are derived from timestamps within the traces. To reflect how the application behaviors impact the effectiveness of the schemes, we assume there is only one application rank per node. In this way, the destination sequence seen by every local circuit runtime is a reflection of single-rank communication behavior. The simulation assumes a 256-node non-blocking optical network, and hence 256 ranks for the applications, except LULESH, which consists of 125 ranks due to a three-dimension decomposition requirement of the problem. Although a non-blocking optical network [41, 42] represents an ideal network scenario, in this paper we only use it as a platform to facilitate an initial comparison of different circuit management strategies. Furthermore, the proposed application-guided prefetching methodology is migratable to blocking networks and our future work will include research in this direction.

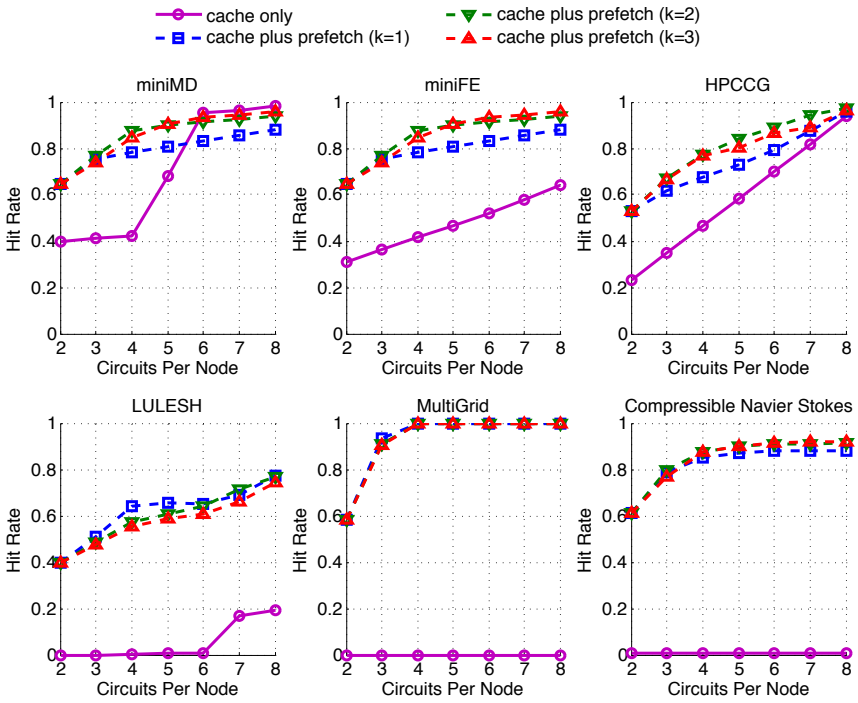

Fig. 2. Circuit hit rates achieved by the caching-plus-prefetch scheme (*dashed*) with different tail lengths k, versus the caching-only scheme (*solid*), in a 256-rank simulation. Both schemes employ the *least recently used* replacement policy.

Table 1 provides a brief description of the benchmarks as well as information regarding their *neighbor-to-rank ratios* and *reuse distances*. Here, the *neighbor-to-rank ratio* is a ratio of the average number of communication neighbors per rank to the number of ranks, indicating a diversity of communication destinations. The *reuse distance* represents how often a destination is re-addressed by a source [12]. Consider a destination sequence from a single source s. If the number of destinations interposed in the sequence between two requests for a destination t is d, then d is the reuse distance of destination t in view of source s. Each application has a "maximum-likelihood" set of reuse distances. These reuse distances are binned logarithmically by powers of 2 in Table 1, showing the most common reuse distances.

4.2 Simulation Results

Hit Rate Fig. 2 shows the hit rate performance of the caching-only and the caching-plus-prefetch schemes with different tail lengths ($k = 1, 2, 3$), against various numbers of circuits per node ($p = 2, 3, ..., 8$). It should be noted that in

case p is smaller than k, the circuit management runtime would fetch no more than p most probable circuits out of the k predicted ones.

Compared to the caching-only scheme, the caching-plus-prefetch scheme significantly increases the circuit hit rate in all of the applications. In applications such as HPCCG, miniFE and miniMD, the hit rate increase is as high as 40%. This number is even larger for LULESH (60%), Multigrid (90%) and CNS (80%). Regarding when the maximal enhancement is achieved by the prefetching scheme, a difference exists between the high-communication-degree (long-reuse-distance) and modest-communication-degree (modest-reuse-distance) applications. Here, we refer the communication degree to the average number of communication neighbors per node, which is proportional to the neighbor-to-rank ratio listed in Table 1.

For modest-communication-degree (modest-reuse-distance) applications, such as HPCCG and miniMD, the maximal hit rate enhancement by the prefetching scheme occurs at small circuit provision numbers – for example, this provision number is two circuits per node for HPCCG and four for miniMD. As the number of provisioned circuits increases, the hit rate of the caching-only scheme grows fast because the "circuit cache size" becomes large enough to cover most of the modest reuse distances, narrowing the performance difference from the prefetching scheme.

For high-communication-degree (long-reuse-distance) applications, such as LULESH, Multigrid and CNS, provisioning more circuits improves the hit rate by a limited amount in the caching-only case. The reason for the limited improvement is that the number of circuits provisioned in the simulation, which is two to eight per node, is still smaller than the communication degree or the most probable reuse distances of the application. Take Multigrid for example, it has a communication degree of 37 (out of the 256 ranks) and a most probable reuse distance in the range of [32, 64), both of which are significantly larger than the maximum number of circuit provisioned. Therefore, the circuit cache size is not large enough to cover the communication degree or reuse distance, resulting in limited hit rate improvement in the caching-only case.

In comparison, the prefetching scheme does not require a large circuit cache size, a small communication degree, or a short reuse distance. The principle of the prefetching scheme merely relies on the existence of correlated communications. In this sense, the prefetch scheme is capable of creating circuit hits even when the number of circuit is small compared to the communication degree or reuse distance. See, for example, the hit rate of Multigrid and CNS in Fig. 2, where the prefetching scheme improves the hit rate from almost 0% to almost 100% compared to the caching-only scheme. Such capability is extremely important for a wide range of applications, 1) which by problem definition have a large communication degree, or 2) whose communication degree increases significantly as the parallelism degree scales. This also means that the prefetching scheme is more scalable than the caching-only scheme.

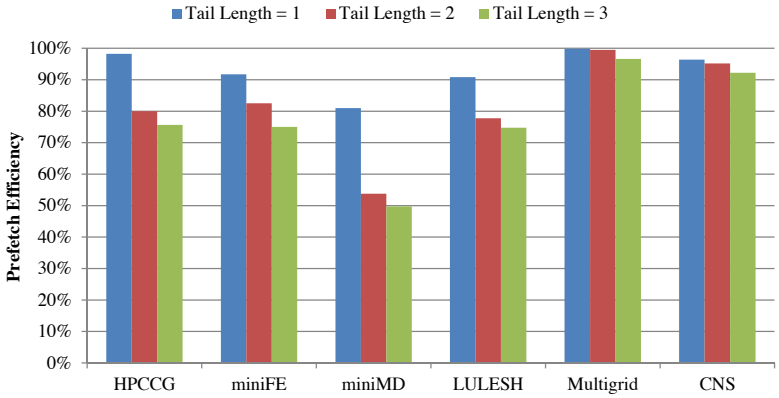

Fig. 3. Prefetch efficiency against different tail lengths k. The prefetch efficiency is a percentage of prefetches that result in a hit, out of the total prefetches.

Tradeoff of Tail Length The tail length parameter determines the number of *followers* to be learned by the predictor and the number of circuits to be prefetched when a *predecessor* is recognized. Although a longer tail length, namely, more prefetches, can generally lead to a higher hit rate, as Fig. 2 shows, it can also increase the reconfiguration cost. We investigate such a tradeoff by evaluating the *prefetch efficiency* under different tail lengths. The *prefetch efficiency* is a percentage of prefetches resulting into a hit out of the total number of prefetches. Fig. 3 captures the prefetch efficiency when the number of circuits per node is three. While the efficiency is in general high for all applications when the tail length is 1, the result also shows that increasing the tail length may decrease the prefetch efficiency, by different extents and based on the applications. Such decrease depends on the length of typical *predecessor-follower* patterns in the application, as well as the repeat frequency of the learned patterns defined by the selected tail length.

Latency Message latencies of the two approaches are compared in Fig. 4. The simulation assumes a reconfiguration latency of 100 ns and a link bandwidth of 100 Gb/s. The reconfiguration latency assumed includes the time to release the old circuit (tens of ns), the time to switch the lightpath (tens of ns [43]), and the time for the hand-shake of the new circuit (tens of ns). As the results show, the prefetch approach significantly reduces the average message latency thanks to a reduced missed rate. In applications such as miniFE and miniMD, a maximal latency reduction of 65% is achieved (at four circuits per node). In Multigrid, an even higher reduction of 90% is achieved (at five circuits per node). These results show that the proposed prefetching scheme can effectively hide the setup latency from application-oriented message communications.

We expect that the above latency-hiding capability would be especially useful to small-message communication in high-bandwdith optical networks. Consider

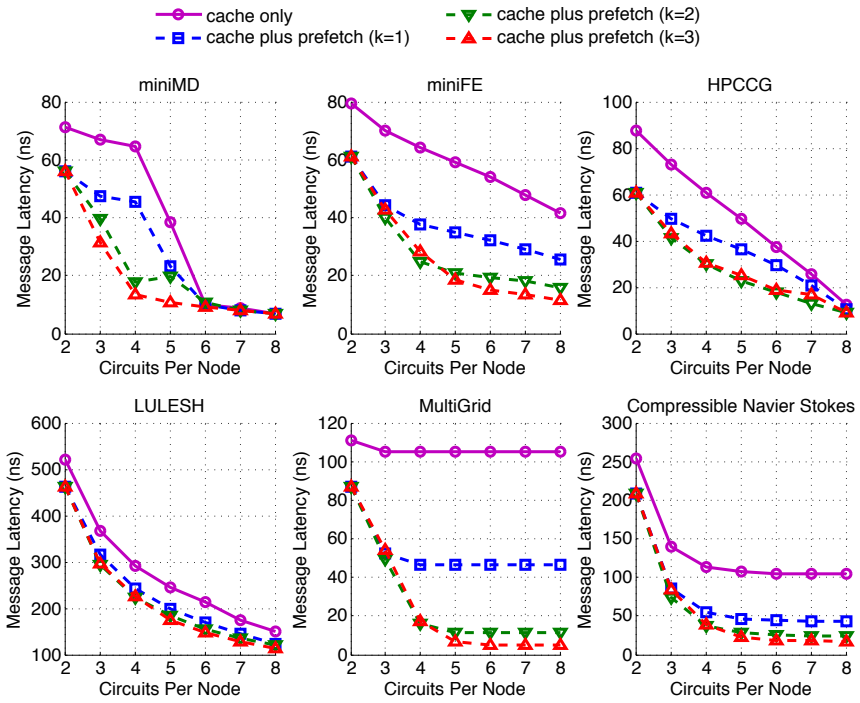

Fig. 4. Message latency of caching-plus-prefetch scheme (*dashed*) with different tail lengths k, versus the caching-only scheme (*solid*), in a 256-rank simulation. Both schemes use the *least recently used* replacement policy. Circuit setup latency is 100 ns, circuit bandwidth is 100 Gb/s.

a small message of 1KB, its transmission time is $1024 \times 8/100 = 81.92$ ns under the assumed link bandwidth (100Gb/s). Such a transmission time is even shorter than the circuit reconfiguration time, which makes hiding the reconfiguration latency even more important for the small-message case.

5 Conclusion

In this work, we build on the circuit-cached architecture by proposing a circuit prefetch approach, to actively avoid the circuit setup latency of optical interconnects in a HPC context. We show a prefetch predictor that learns application-specific *predecessor-follower* patterns from the destination sequence, with a variable control on the length of such patterns. Simulation results based on a wide spectrum of scientific applications, which represent various communication degrees and circuit reuse distances, show that the proposed prefetching scheme significantly improves the circuit hit rate over the caching-only scheme. The prefetching scheme is also shown to effectively hide the reconfiguration latency

from application communication, which capability is important for small-message scenarios.

The current work is only an introductory exploration of reconfiguration in optical interconnection networks providing end-to-end communication circuits. Many issues deserve further study. A high circuit hit rate may not directly translate into a short execution time for the application. More detailed system-level simulations at larger scale could further explore the performance impacts, particularly under more detailed circuit NIC and switch contention models. The current work does not follow the popular MPI-everywhere model with one rank per core. Future study could explore the tradeoffs involved in increasing the number of MPI ranks per node. However, the current work should be considered part of a co-design process, exploring how new technologies might tip the design point for balancing distributed- and shared-memory parallelism.

Acknowledgments. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

References

1. Ophir, Noam, Christopher Mineo, David Mountain, and Keren Bergman. "Silicon photonic microring links for high-bandwidth-density, low-power chip I/O." Micro, IEEE 33, no. 1 (2013): 54-67.
2. Doany, Fuad E., Benjamin G. Lee, Daniel M. Kuchta, Alexander V. Rylyakov, Christian Baks, Christopher Jahnes, Frank Libsch, and Clint L. Schow. "Terabit/Sec VCSEL-based 48-channel optical module based on holey CMOS transceiver IC." Lightwave Technology, Journal of 31, no. 4 (2013): 672-680.
3. Doany, Fuad E., Benjamin G. Lee, Clint L. Schow, Cornelia K. Tsang, Christian Baks, Young Kwark, Richard John, John U. Knickerbocker, and Jeffrey A. Kash."Terabit/s-class 24-channel bidirectional optical transceiver module based on TSV Si carrier for board-level interconnects." In Electronic Components and Technology Conference (ECTC), 2010 Proceedings 60th, pp. 58-65. IEEE, 2010.
4. Schares, Laurent, Jeffrey A. Kash, Fuad E. Doany, Clint L. Schow, Christian Schuster, Daniel M. Kuchta, Petar K. Pepeljugoski et al. "Terabus: Terabit/second-class card-level optical interconnect technologies." Selected Topics in Quantum Electronics, IEEE Journal of 12, no. 5 (2006): 1032-1044.
5. Shalf, John, Shoaib Kamil, Leonid Oliker, and David Skinner. "Analyzing ultra-scale application communication requirements for a reconfigurable hybrid interconnect." In Proceedings of the 2005 ACM/IEEE conference on Supercomputing, p. 17. IEEE Computer Society, 2005.
6. Barker, Kevin J., Alan Benner, Ray Hoare, Adolfy Hoisie, Alex K. Jones, Darren K. Kerbyson, Dan Li et al. "On the feasibility of optical circuit switching for high performance computing systems." In Proceedings of the 2005 ACM/IEEE conference on Supercomputing, p. 16. IEEE Computer Society, 2005.
7. Caulfield, H. John, and Shlomi Dolev. "Why future supercomputing requires optics." Nature Photonics 4, no. 5 (2010): 261-263.

8. Taubenblatt, Marc A. "Optical interconnects for high-performance computing." Journal of Lightwave Technology 30, no. 4 (2012): 448-457.
9. Biberman, Aleksandr, and Keren Bergman. "Optical interconnection networks for high-performance computing systems." Reports on Progress in Physics 75, no. 4 (2012): 046402.
10. Padmaraju, Kishore, Lian-Wee Luo, Xiaoliang Zhu, Madeleine Glick, Raj Dutt, Michal Lipson, and Keren Bergman. "Wavelength locking of a wdm silicon microring demultiplexer using dithering signals." In Optical Fiber Communication Conference, pp. Tu2E-4. Optical Society of America, 2014.
11. Padmaraju, Kishore, and Keren Bergman. "Resolving the thermal challenges for silicon microring resonator devices." Nanophotonics 3, no. 4-5 (2014): 269-281.
12. Wen, Ke, David Calhoun, Sébastien Rumley, Xiaoliang Zhu, Yang Liu, Lian Wee Luo, Ran Ding et al. "Reuse Distance Based Circuit Replacement in Silicon Photonic Interconnection Networks for HPC." In High-Performance Interconnects (HOTI), 2014 IEEE 22nd Annual Symposium on, pp. 49-56. IEEE, 2014.
13. Zheng, Xuezhe, Dinesh Patil, Jon Lexau, Frankie Liu, Guoliang Li, Hiren Thacker, Ying Luo et al. "Ultra-efficient 10Gb/s hybrid integrated silicon photonic transmitter and receiver." Optics Express 19, no. 6 (2011): 5172-5186.
14. Zheng, Xuezhe, Frankie Liu, Jon Lexau, Dinesh Patil, Guoliang Li, Ying Luo, Hiren Thacker et al. "Ultra-low power arrayed CMOS silicon photonic transceivers for an 80 Gbps WDM optical link." In National Fiber Optic Engineers Conference, p. PDPA1. Optical Society of America, 2011.
15. Streshinsky, Matthew, Ari Novack, Ran Ding, Yang Liu, Andy Eu-Jin Lim, Patrick Guo-Qiang Lo, Tom Baehr-Jones, and Michael Hochberg. "Silicon Parallel Single Mode 48 50 Gb/s Modulator and Photodetector Array." Journal of Lightwave Technology 32, no. 22 (2014): 3768-3775.
16. Fukazawa, Tatsuhiko, Fumiaki Ohno, and Toshihiko Baba. "Very compact arrayed-waveguide-grating demultiplexer using Si photonic wire waveguides." Japanese journal of applied physics 43, no. 5B (2004): L673.
17. Fang, Qing, Tsung-Yang Liow, Jun Feng Song, Kah Wee Ang, Ming Bin Yu, Guo Qiang Lo, and Dim-Lee Kwong. "WDM multi-channel silicon photonic receiver with 320 Gbps data transmission capability." Optics express 18, no. 5 (2010): 5106-5113.
18. Zheng, Xuezhe, Ivan Shubin, Guoliang Li, Thierry Pinguet, Attila Mekis, Jin Yao, Hiren Thacker et al. "A tunable 1x4 silicon CMOS photonic wavelength multiplexer/demultiplexer for dense optical interconnects." Optics express 18, no. 5 (2010): 5151-5160.
19. Horst, Folkert, William MJ Green, Solomon Assefa, Steven M. Shank, Yurii A. Vlasov, and Bert Jan Offrein. "Cascaded Mach-Zehnder wavelength filters in silicon photonics for low loss and flat pass-band WDM (de-) multiplexing." Optics express 21, no. 10 (2013): 11652-11658.
20. Piggott, Alexander Y., Jesse Lu, Konstantinos G. Lagoudakis, Jan Petykiewicz, Thomas M. Babinec, and Jelena Vukovi. "Inverse design and demonstration of a compact and broadband on-chip wavelength demultiplexer." Nature Photonics 9, no. 6 (2015): 374-377.
21. Luo, Lian-Wee, Noam Ophir, Christine P. Chen, Lucas H. Gabrielli, Carl B. Poitras, Keren Bergmen, and Michal Lipson. "WDM-compatible mode-division multiplexing on a silicon chip." Nature communications 5 (2014).
22. Luo, Lian-Wee, Noam Ophir, Christine Chen, Lucas H. Gabrielli, Carl B. Poitras, Keren Bergman, and Michal Lipson. "Simultaneous mode and wavelength division multiplexing on-chip." arXiv preprint arXiv:1306.2378 (2013).

23. Chen, Christine P., Jeffrey B. Driscoll, Richard R. Grote, Brian Souhan, Richard M. Osgood, and Keren Bergman. "Mode and Polarization Multiplexing in a Si Photonic Chip at 40Gb/s Aggregate Data Bandwidth." Photonics Technology Letters, IEEE 27, no. 1 (2015): 22-25.
24. Stern, Brian, Xiaoliang Zhu, Christine Chen, Lawrence Tzuang, Jaime Cardenas, Keren Bergman, and Michal Lipson. "Integrated Switch for Mode-Division Multiplexing (MDM) and Wavelength-Division Multiplexing (WDM)." In CLEO: Science and Innovations, pp. STh1F-2. Optical Society of America, 2015.
25. Wang, Jian, Pengxin Chen, Sitao Chen, Yaocheng Shi, and Daoxin Dai. "Improved 8-channel silicon mode demultiplexer with grating polarizers." Optics express 22, no. 11 (2014): 12799-12807.
26. Rumley, Sébastien, Dessislava Nikolova, Robert Hendry, Qi Li, David Calhoun, and Keren Bergman. "Silicon photonics for exascale systems." Journal of Lightwave Technology 33, no. 3 (2015): 547-562.
27. Shacham, Assaf, Benjamin A. Small, Odile Liboiron-Ladouceur, and Keren Bergman. "A Fully Implemented 12 12 Data Vortex Optical Packet Switching Interconnection Network." Journal of Lightwave Technology 23, no. 10 (2005): 3066.
28. Shacham, Assaf, and Keren Bergman. "Building ultralow-latency interconnection networks using photonic integration." Micro, IEEE 27, no. 4 (2007): 6-20.
29. Hemenway, Roe, Richard Grzybowski, Cyriel Minkenberg, and Ronald Luijten. "Optical-packet-switched interconnect for supercomputer applications [Invited]." Journal of Optical Networking 3, no. 12 (2004): 900-913.
30. Yang, Yuanyuan, Jianchao Wang, and Yi Pan. "Permutation capability of optical multistage interconnection networks." Journal of parallel and Distributed computing 60, no. 1 (2000): 72-91.
31. Hendry, Gilbert. "Decreasing network power with on-off links informed by scientific applications." In Parallel and Distributed Processing Symposium Workshops and PhD Forum (IPDPSW), 2013 IEEE 27th International, pp. 868-875. IEEE, 2013.
32. Jouppi, Norman P. "Improving direct-mapped cache performance by the addition of a small fully-associative cache and prefetch buffers." In Computer Architecture, 1990. Proceedings., 17th Annual International Symposium on, pp. 364-373. IEEE, 1990.
33. Mowry, Todd C., Monica S. Lam, and Anoop Gupta. "Design and evaluation of a compiler algorithm for prefetching." In ACM Sigplan Notices, vol. 27, no. 9, pp. 62-73. ACM, 1992.
34. Cao, Pei, Edward W. Felten, Anna R. Karlin, and Kai Li. A study of integrated prefetching and caching strategies. Vol. 23, no. 1. ACM, 1995.
35. Heroux, Michael A., Douglas W. Doerfler, Paul S. Crozier, James M. Willenbring, H. Carter Edwards, Alan Williams, Mahesh Rajan, Eric R. Keiter, Heidi K. Thornquist, and Robert W. Numrich. "Improving performance via mini-applications." Sandia National Laboratories, Tech. Rep (2009).
36. Numrich, Robert W., and Michael A. Heroux. "A performance model with a fixed point for a molecular dynamics kernel." Computer Science-Research and Development 23, no. 3-4 (2009): 195-201.
37. Karlin, Ian, Abhinav Bhatele, Bradford L. Chamberlain, Jonathan Cohen, Zachary Devito, Maya Gokhale, Riyaz Haque et al. "Lulesh programming model and performance ports overview." Lawrence Livermore National Laboratory (LLNL), Livermore, CA, Tech. Rep (2012).
38. Karlin, Ian, Abhinav Bhatele, Jeff Keasler, Bradford L. Chamberlain, Jonathan Cohen, Zachary DeVito, Riyaz Haque et al. "Exploring traditional and emerging

parallel programming models using a proxy application." In Parallel and Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on, pp. 919-932. IEEE, 2013.
39. http://portal.nersc.gov/project/CAL/exact.htm
40. Wilke, Jeremiah J. and Kenny, Joseph P. "Using discrete event simulation for programming model exploration at extreme-scale: Macroscale components for the structural simulation toolkit (SST)." Sandia National Laboratories, Tech. Rep. SAND2015-1027.
41. Petracca, Michele, Benjamin G. Lee, Keren Bergman, and Luca P. Carloni. "Design exploration of optical interconnection networks for chip multiprocessors." In High Performance Interconnects, 2008. HOTI'08. 16th IEEE Symposium on, pp. 31-40. IEEE, 2008.
42. Sherwood-Droz, Nicols, Howard Wang, Long Chen, Benjamin G. Lee, Aleksandr Biberman, Keren Bergman, and Michal Lipson. "Optical 4x4 hitless slicon router for optical networks-on-chip (NoC)." Optics express 16, no. 20 (2008): 15915-15922.
43. Calhoun, David, Ke Wen, Xiaoliang Zhu, Sébastien Rumley, L. Luo, Y. Liu, R. Ding et al. "Dynamic reconfiguration of silicon photonic circuit switched interconnection networks." In Proc. IEEE High Perform. Extreme Comput. Conf. 2014.