Performance studies of 3D-Hyper-FleX-LION for HPC applications

Performance Studies of 3D-Hyper-FleX-LION for HPC Applications

Zhiyan Liu, Roberto Proietti*, Xiaoliang Chen, and S.J.Ben Yoo*

Department of Electrical and Computer Engineering, University of California, Davis, California 95616, USA

**rproietti@ucdavis.edu, sbyoo@ucdavis.edu*

Abstract: This paper studies the performance of 3D-Hyper-FleX-LION for HPC systems. The simulation results obtained for different HPC applications (i.e. Fill Boundary, Crystal Router, MiniFE, and MiniDFT) show up to 2.8× improvements in throughput per watt when compared with a Fat-Tree with no oversubcription.

1. Introduction

As High Performance Computing (HPC) systems are entering the beyond-Exascale Era, the scalability, latency, and energy efficiency of their interconnection networks are key to the system performance. In fact, many applications exhibit irregular communication patterns and have execution phases that can be highly affected by hotspots in the network. Most of the supercomputers in the top500 list leverage electronic packet switches (EPS) wired based on Fat-Tree or Torus-based topologies [1] with rigid optical point-to-point connectivity for inter-rack communications. These multi-hop topologies directly impact the zero-load latency, which is significant to system performance as switch traversal latencies can be on the order of 100's of nanoseconds. Making efficient use of network bandwidth is another critical consideration for system performance and power consumption. As communication patterns between compute nodes are typically not evenly distributed and show high variances in the link utilization (between low-utilization computing-intense and high-utilization communication-intense phases), it is challenging to provide sufficient bandwidth for high-communication phases between specific node pairs without wasting energy in low-utilization phases. Recently, there has been a significant attention to design low-diameter architectures augmented with photonic switching to elastically assign the links bandwidth based on the traffic shape [2,3]. In [3], we proposed an optical and elastic Hyper-X network that leverages the reconfigurable all-to-all property of SiPh Flex-LIONS switches [4] to provide a scalable, energy-efficient, and low-diameter elastic interconnection network. This paper investigates and demonstrates the benefits of 3D-Hyper-FleX-LION for different HPC applications [5] and with different network oversubscription ratios.

2. 3D-Hyper-FleX-LION HPC Architecture

Figure 1(a) shows the working principle and block diagram of Flex-LIONS (Flexible Low-Latency Interconnect Optical Network Switch) [4], which is composed of an $N \times N$ arrayed waveguide grating router (AWGR) and an $N \times N$ colorless optical switch placed in parallel between b-port microring resonator (MRR)-based wavelength selective switches. When the MRRs are set off-resonance, each input port of the AWGR receives N wavelengths, achieving all-to-all interconnection (one dedicated wavelength per node pair) based on the well-known wavelength-routing property of an AWGR. Alternatively, the MRRs can be tuned in resonance to break the all-to-all connectivity and increase the number of wavelengths between specific node pairs by routing certain wavelengths through the colorless optical switch. Figure 1(a) shows an example where the bandwidth between node 1 and 2 (node 7 and 8) increases by a factor of 3 (2). Note that the reconfiguration operation allows to reconfigure the bandwidth only among $N/2$ disjoint node pairs (this constraint is referred to as w/ C in the simulations reported in Section 4). Figure 1(b) shows the 3D-Hyper-FleX-LION architecture (also FleX-LION) with identical k-port top-of-rack (ToR) switches. Each ToR uses a group of ports (e.g $k/2$ or $k/4$ for oversubscription ratio of $1:1$ or $3:1$, respectively) for server connection. The remaining ToR ports are partitioned into three groups used for intra-pod and inter-pod communication. Each port in each group drives a WDM optical transceiver for connection to the corresponding Flex-LIONS. As reported in details in [3], the architecture can support $k^4/512$ ($k^4/256$) hosts when each ToR uses $k/2$ ($3k/4$) ports for inter-rack communication. This is equal or larger than what a Fat-Tree without oversubscription can support ($k^3/4$) when the switch ASIC radix is equal or larger than 128 (64).

3. Reconfiguration Algorithm

This section describes the algorithm and the principle of reconfiguration used to adapt the intra and inter pod interconnection as a function of the traffic profile of different HPC applications [5]. Here we assume an application-driven scenario (see Figure 2) with reconfiguration done prior to the execution of an application. To calculate the graph, we used the following algorithm. We denote $T \in \mathbb{R}^{N \times N}$ the estimated traffic matrix of a $N \times N$ switch,

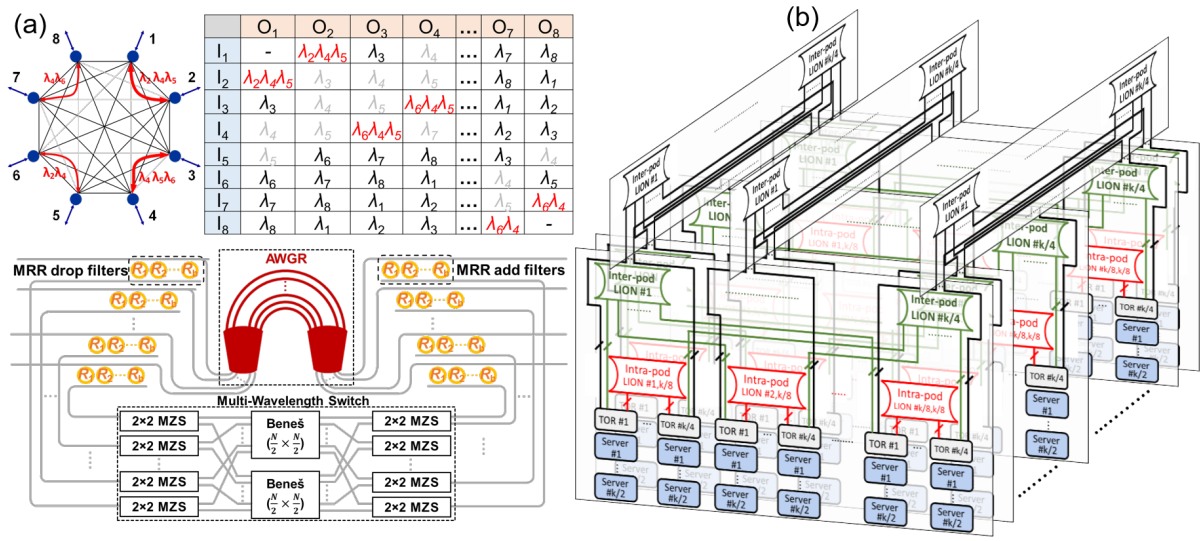

Fig. 1: (a) Working principle and block diagram of Flex-LIONS. (b) 3D-Hyper-FleX-LION architecture [3]

where T_{ij} denotes the traffic rate from the i-th input to the j-th output in Gbps. $\mathscr{D} = \{D^{(1)}, D^{(2)}, ..., D^{(n)}\}$ denotes the set of all possible logical topologies satisfying certain constraints, where $D^{(k)} \in \mathbb{Z}^{N\times N}$ is a symmetric matrix and $D_{ij}^{(k)}$ denotes the number of links between the i-th input and the j-th output after reconfiguration. The set $\mathscr{D}$ can be found using depth-first search. Next, we normalize all matrices to have the same ℓ_1-norm by assigning $\hat{T} = \frac{T}{\|T\|_1}$ and $\hat{D}^{(k)} = \frac{D^{(k)}}{\|D^{(k)}\|_1}$ for $k = 1, 2, \ldots, n$. The optimal topology is then found by picking $\hat{D}^{(k)}$ with the minimum mean square error compared to $\hat{T}$. The procedure is summarized in Algorithm 1.

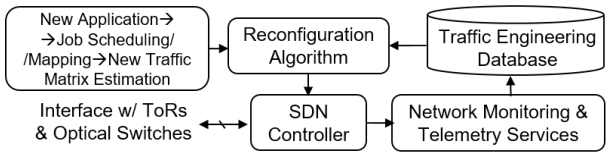

Fig. 2: Application-driven reconfiguration flow. When a new application job arrives, the job is scheduled and mapped to the network and a new traffic matrix is estimated and input to the algorithm. The output topology is input to the SDN controller that interfaces with ToRs and Flex-LIONSs for reconfiguration.

Architecture	Fat Tree	FleX-LION	Fat Tree	FleX-LION
Radix	14	24	10	16
Oversubscription	1 : 1	1 : 1	1 : 1	3 : 1
Num. of: Servers, ASICs	686, 244	648, 54	250, 125	256, 64
Num. of: LIONSs, TRXs	0, 2744	45, 648	0, 1000	48, 768

Table 1: The four architectures evaluated in the simulations.

Algorithm 1: Search for the optimal connectivity graph

Initialize $\mathscr{D}$ with depth-first search;
$\hat{T} \leftarrow \frac{T}{\|T\|_1}$; $e_{\min} \leftarrow \infty$;
for $D^{(k)} \in \mathscr{D}$ **do**
  $\hat{D}^{(k)} = \frac{D^{(k)}}{\|D^{(k)}\|_1}$;
  $e = \sum_{i=1}^{N}\sum_{j=1}^{N} |\hat{T}_{ij} - \hat{D}_{ij}^{(k)}|^2$;
  if $e < e_{min}$ **then**
    $e_{\min} = e$; $D = D^{(k)}$;
  end
end
return D;

4. Performance Simulation Results

We used NetBench [6] for packet-level performance evaluation studies. All links are homogeneous with capacity of 10 Gbps and link delay of 20 ns. Each port has a buffer size of 150,000 bytes (packets are dropped if the buffers get full). The flow arrival events are generated with a Poisson process with parameter λ, i.e. the average flow arrival rate. At each arrival event, one source-destination pair is picked from a traffic probability distribution. Fig. 3 (first column) shows the four traffic distributions derived from real HPC applications (i.e. Fill Boundary, Crystal Router, MiniFE and MiniDFT [5]). All the simulations use equal-cost multi-path (ECMP) without flow splitting as the routing scheme. We considered two FleX-LION architectures with different oversubscription ratios (see Table 1) and compared them with Fat-Tree networks that support a similar number of servers. In the first case, we considered a FleX-LION network with $k = 24$ and $1 : 1$ oversubscription. Each ToR allocates 12 ports for server connection, and the other half are used for intra-pod and inter-pod communications. Nine 6×6 intra-pod Flex-LIONSs are reconfigurable, while all other 3×3 switches are AWGRs without reconfiguration capability given the small radix required at this scale for the second and third dimensions. In the second case, we considered a FleX-LION network with $k = 16$ and $3 : 1$ oversubscription, where each ToR allocates four ports for servers and the remaining ports for intra-pod and inter-pod communications. All 54 switches are 4×4 Flex-LIONSs. Fig. 3 shows the average packet latency and throughput per watt of Fat Tree, FleX-LION without reconfiguration

(FleX-LION w/o R), with constrained reconfiguration (FleX-LION w/ C) and with unconstrained reconfiguration (FleX-LION w/o C).

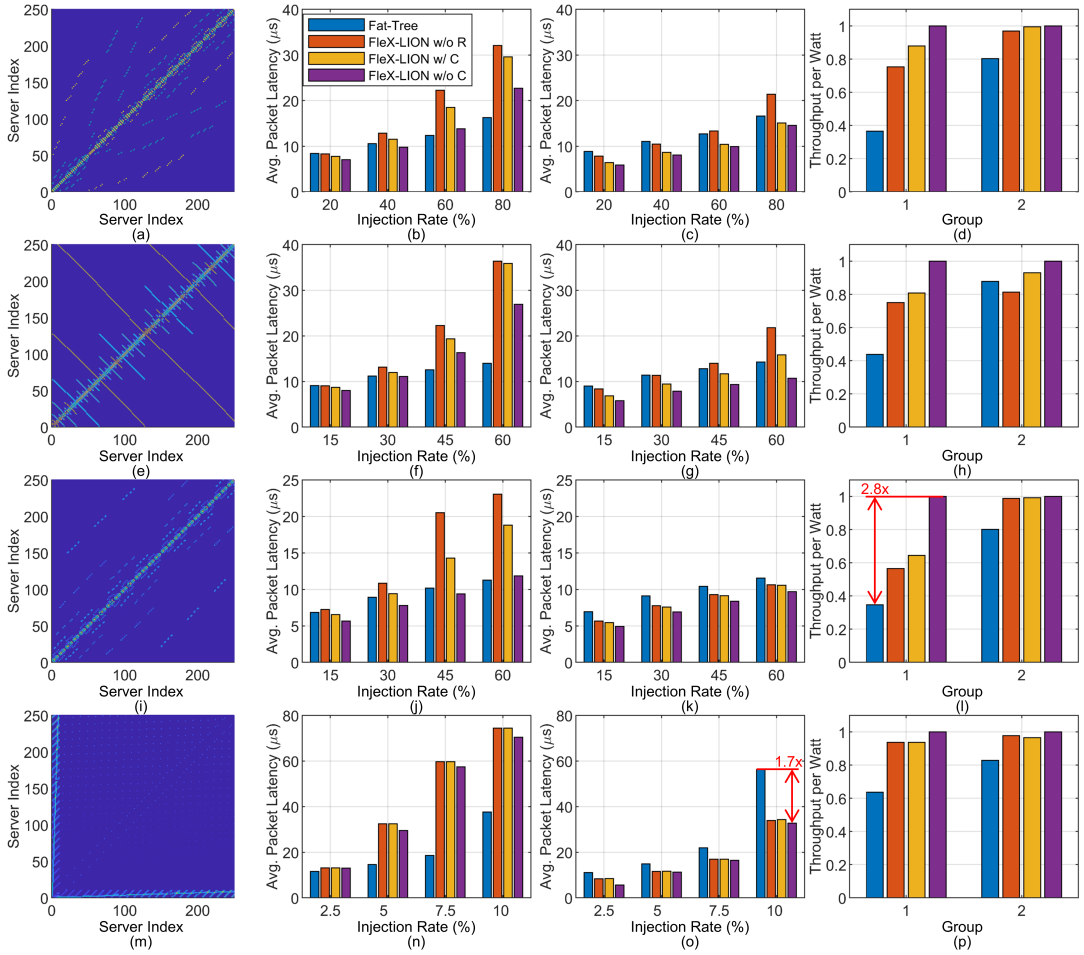

Fig. 3: Average packet latency of the 1:1 oversubscribed architecture (second column) and the 3:1 oversubscribed architecture (third column) under Fill Boundary ((a)-(c)), Crystal Router ((e)-(g)), MiniFE ((i)-(k)), and MiniDFT ((m)-(o)). The fourth column shows the normalized throughput per watt performance for 1:1 oversubscription (group 1) and 3:1 oversubscription (group 2). The throughput is measured at packet loss rate of 1%. The power consumption is calculated with the number of ASICs, LIONSs and optical TRXs based on the power numbers reported in [3].

The results show that the proposed architecture provides up to 1.7× latency reduction when compared to a Fat-Tree network. The reconfiguration leads to significant throughput per watt gain (up to 2.8×), especially in traffic traces with rich intra-pod traffic. It is worth noting that the 1:1 oversubscribed architecture performed worse than Fat Tree in terms of packet latency under heavier workloads. This can be attributed to the fact that the reconfiguration capability is limited to intra-pod switches under this setting.

5. Conclusions

We demonstrated that a 3D-Hyper-FleX-LION architecture could achieve lower latency and higher energy efficiency than a Fat-Tree network by adapting the links bandwidth to the traffic characteristics of the HPC applications. Future work will look into dynamic reconfiguration at run-time and related challenges.

References

1. ”Top500”, https://www.top500.org/.
2. M. Y. Teh et al., ”Flexspander: augmenting expander networks in high-performance systems with optical bandwidth steering,” IEEE/OSA Journal of Optical Communications and Networking, vol. 12, no. 4, pp. B44-B54, 2020.”
3. G. Liu et al., ”Architecture and performance studies of 3D-Hyper-FleX-LION for reconfigurable all-to-all HPC networks,” in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC ’20). IEEE Press, Article 26, 1–16.
4. X. Xiao et al., ”Silicon Photonic Flex-LIONS for Bandwidth-Reconfigurable Optical Interconnects,” IEEE Journal of Selected Topics in Quantum Electronics, vol. 26, no. 2, pp. 1-10, 2020.
5. ”Characterization of the DOE Mini-apps”, https://portal.nersc.gov/project/CAL/doe-miniapps.htm.
6. ”NetBench”, https://github.com/ndal-eth/netbench.

This work was supported in part by ARO award # W911NF1910470, DoD award # H98230-19-C-0209, NSF ECCS award # 1611560, and by DoE UAI consortium award # DE-SC0019582, DE-SC0019526, and DE-SC001969.