Understanding the Impact of Fat-Tree Network Locality on Application Performance

Philip Taffet
Rice University
Houston, Texas
ptaffet@rice.edu

Ian Karlin (Advisor)
Edgar A. Leon (Advisor)
Lawrence Livermore National Lab
Livermore, California
karlin1,leon@llnl.gov

John Mellor-Crummey (Advisor)
Rice University
Houston, Texas
johnmc@rice.edu

ACM Reference Format:
Philip Taffet, Ian Karlin (Advisor), Edgar A. Leon (Advisor), and John Mellor-Crummey (Advisor). 2017. Understanding the Impact of Fat-Tree Network Locality on Application Performance. In *Proceedings of The International Conference for High Performance Computing, Networking, Storage and Analysis, Denver, CO, November 12–17, 2017 (SC17),* 2 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Network congestion can be a significant cause of performance loss and variability for many message passing programs. However, few studies have used a controlled environment with virtually no other extraneous sources of network traffic to observe the impact of application placement and multi-job interactions on overall performance. In this work, we study the impact of network placement and task mapping on the performance of three DOE applications. We observe that for the studied applications and job sizes, the impact of congestion is fairly minimal. In addition, in most cases, the cyclic MPI task mapping strategy increases performance despite also increasing total network traffic.

2 METHODOLOGY

2.1 Applications

We chose three applications for this study (Table 1). AMG and UMT are proxy applications part of the CORAL procurement suite [1] and pF3D is a production application. All applications were run with one MPI rank per core, as is typical for these applications on commodity clusters at LLNL [3]. Problem sizes were chosen so that each run took 5-7 minutes on 30 nodes (28 nodes for pF3D).

We use the figure of merit reported by UMT and AMG as our performance metric. Since all pF3D runs completed the same amount of work, we used $1/t$ as pF3D's figure of merit, where t is wallclock time excluding I/O. For all figures of merit, higher is better.

2.2 Hardware

All experiments ran on a 186-node partition of Quartz, a cluster at LLNL. Each node contains dual 18-core Intel Xeon E5-2695v4 (Broadwell) sockets. Quartz's interconnect uses 100 Gbps Omni-path arranged in a 3-level fat-tree with a 2:1 taper. The 48-port leaf switches connect to 31 compute nodes, 1 service node, and 16 links to second-level switches. The 6 leaf switches in our partition share the same second-level switches, and no other users were able to run jobs on our partition during these experiments. In addition, network traffic counters show that other users' jobs caused minimal load to the core and second-level switches during the experiments.

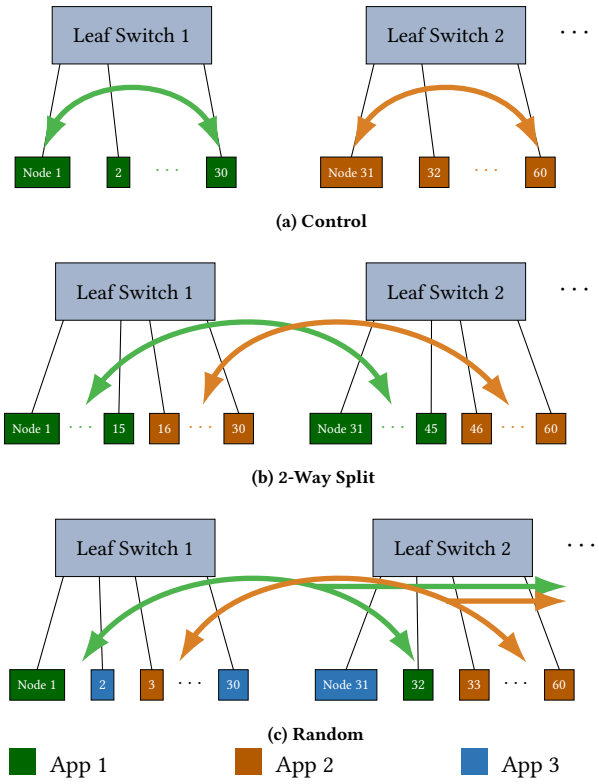

Figure 1: Network placement patterns

2.3 Experiments

We ran each application in several network placement patterns. Each application was run with:

- *Control:* all nodes connected to one leaf switch (Figure 1a).
- *s-Way Split:* $30/s$ nodes connected to each of s leaf switches, for $s = 2, 3, 6$ ($s = 2$ depicted in Figure 1b).
- *Random:* 30 nodes randomly chosen from the available 6 leaf switches (Figure 1c).

All nodes in the partition ran experiments with the same placement pattern simultaneously. For example, in the *2-Way Split*, while 15 nodes on one leaf switch ran AMG, the other 15 ran UMT or pF3D.

SC17, November 12–17, 2017, Denver, CO
2017. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Table 1: Summary of applications and bullies used for this poster. Medium sized messages are 2KB - 32 KB.

Name	Description	P2P message sizes	Collective sizes
AMG (App)	Algebraic multi-grid solver	Large	Small
UMT (App)	Radiation transport	Large	Small
pF3D (App)	Laser-plasma interaction	Small to large	Large
MPIBW (Bully)	MPI bandwidth benchmark	Large	None
MPIPPS (Bully)	MPI message-rate benchmark	Small	None

Table 2: Average figures of merit (as percent deviation from Control) for each application and experiment. Positive values mean a higher figure of merit than Control. Bold green values are statistically significant compared to Control at $\alpha = .01$, and *italic orange values* are significant at $\alpha = .05$.

	Cyclic			Block		
Experiment	AMG	UMT	pF3D	AMG	UMT	pF3D
Split with AMG	-	*	0.0%	-	*	0.4%
Split with UMT	*-0.2%*	-	-0.2%	0.1%	-	0.5%
Split with pF3D	-0.1%	0.1%	-	-0.3%	0.0%	-
Bully MPIBW	**-0.3%**	0.0%	0.2%	0.1%	-0.3%	0.4%
Bully MPIPPS	**-0.2%**	-0.1%	0.0%	0.1%	-0.1%	0.6%
3-Way Split	**-0.2%**	-0.2%	**-1.5%**	*-2.1%*	-0.1%	8.3%
6-Way Split	**-0.3%**	-0.3%	**-0.7%**	*-2.8%*	*-0.4%*	0.4%
Random	**-0.2%**	*-0.4%*	**-1.7%**	-1.5%	-0.1%	5.7%
In the Wild	*-1.6%*	-0.2%	**-1.2%**	*-2.4%*	-2.9%	-0.3%

We also ran two special experiments. Each application was run:

- *With Bully:* in the *2-Way Split* configuration, while the other 15 nodes from each switch flooded the network with congestion by running a bandwidth (MPIBW) or packet rate (MPIPPS) benchmark.
- *In The Wild:* a normally scheduled allocation, after the dedicated access time.

All placement patterns except *In The Wild* were repeated 3 times using the MPI block mapping and 3 times using cyclic mapping [2]. To account for additional variance due to the scheduler, *In The Wild* placements were repeated 5 times for each mapping policy.

3 CONCLUSIONS AND FUTURE WORK

The strongest and most consistent trend is that cyclic task mapping performs better than block, which is shown in Table 3. Interestingly, network counters during the experiments show that using the cyclic task mapping increases the network traffic. We are still investigating this surprising result.

Table 3: The relative performance difference between block and cyclic task mappings. Positive values mean cyclic performed better than block. Bold green values are statistically significant at $\alpha = .01$

Experiment	AMG	UMT	pF3D
Control	**3.5%**	**1.5%**	**9.8%**
6-Way Split	**6.0%**	**1.6%**	**8.7%**
Random	**4.9%**	**1.2%**	2.5%
In the Wild	**4.4%**	4.2%	**8.9%**

Overall, the results indicate that the fat-tree interconnect on Quartz is robust. All applications, especially AMG and UMT, were minimally or negligibly affected by congestion and poor placement, relative to total application runtime. The most network-heavy application, pF3D, was the most sensitive to placement, congestion, and task mapping. Yet, its greatest deviations were fairly minimal in most cases, especially with cyclic task mapping (Table 2). Furthermore, for UMT and pF3D, the results from *In The Wild* experiments are between the best and worst controlled placement policies, confirming that normally scheduled jobs tend to receive allocations of average quality. The mean for AMG's *In The Wild* experiment is noticeably worse than other placement policies. However, the results are bimodal; three of the five runs were within 0.3% of *Control*, while the other two were about 6% slower. AMG's sensitivity to system noise [3] may explain this result.

These experiments provide evidence that locality-oblivious schedulers may be suitable for systems with fat-tree interconnects. The small performance gains do not appear to offset the increased job queue times and reduced system utilization from additional locality constraints. This is in contrast to torus networks, for which locality-based schedulers improve runtime and system throughput [4].

We plan to extend this work by trying larger job sizes and additional applications. We would also like to re-run *In The Wild* placements without any congesting traffic to isolate two possible causes of performance impact: congestion and placement. To better understand the mechanisms by which these two causes result in performance loss, we plan on collecting network traces during some of these experiments and replaying them in a network simulator.

REFERENCES

[1] 2014. CORAL Benchmark Codes. (2014). https://asc.llnl.gov/CORAL-benchmarks/

[2] Hewlett-Packard Development Company 2007. *HP-MPI User's Guide* (11 ed.). Hewlett-Packard Development Company. Manufacturing Part Number: B6060-96024.

[3] Edgar A. León, Ian Karlin, Abhinav Bhatele, Steven H. Langer, Chris Chambreau, Louis H. Howell, Trent D'Hooge, and Matthew L. Leininger. 2016. Characterizing Parallel Scientific Applications on Commodity Clusters: An Empirical Study of a Tapered Fat-tree. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '16)*. IEEE Press, Piscataway, NJ, USA, Article 78, 12 pages. http://dl.acm.org/citation.cfm?id=3014904.3015009

[4] Christopher Zimmer, Saurabh Gupta, Scott Atchley, Sudharshan S. Vazhkudai, and Carl Albing. 2016. A Multi-faceted Approach to Job Placement for Improved Performance on Extreme-scale Systems. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '16)*. IEEE Press, Piscataway, NJ, USA, Article 87, 11 pages. http://dl.acm.org/citation.cfm?id=3014904.3015021

Prepared by LLNL under Contract DE-AC52-07NA27344.