Increasing Fabric Utilization with Job-Aware Routing

[Extended Abstract: Standard Poster]

Jens Domke
Institute of Computer Engineering
TU Dresden, Germany
jens.domke@tu-dresden.de

ABSTRACT

The InfiniBand (IB) technology became one of the most widely used interconnects for HPC systems in recent years. The achievable communication throughput for parallel applications depends heavily on the available number of links and switches of the fabric. These numbers are derived from the quality of the used routing algorithm, which usually optimizes the forwarding tables for global path balancing. However, in a multi-user/multi-job HPC environment this results in suboptimal usage of the shared network by individual jobs. We extend an existing routing algorithm to factor in the locality of running parallel applications, and we create an interface between the batch system and the subnet manager of IB to drive necessary re-routing steps for the fabric. As a result, our job-aware routing allows each running parallel application to make better use of the shared IB fabric, and therefore increase the application performance and the overall fabric utilization.

Categories and Subject Descriptors

C.2.2 [**Network Protocols**]: Routing protocols

General Terms

ALGORITHMS, MANAGEMENT

Keywords

Batch system, InfiniBand, job-aware DFSSSP routing, network utilization

1. INTRODUCTION

Large HPC-systems at national labs and universities are usually used by many users running a diverse set of serial and parallel applications. A common choice in production HPC-systems is to use InfiniBand as high-performance interconnection network [11]. InfiniBand uses static, oblivious routing to forward packets in the fabric [6]. One known problem with static, oblivious routing is a potential mismatch between the traffic pattern and configured paths resulting in degraded throughput [4]. This is especially true for multi-user/multi-job HPC environment. Non-standardized extensions of InfiniBand are emerging and aim to solve these issues via (semi-)adaptive routing [9].

SC15 Austin, Texas, USA

2. JOB-AWARE ROUTING OF IB FABRICS

To circumvent the disadvantages of static routing within an multi-job HPC-system, and as an alternative to adaptive routing, we propose the linkage of the batch system (e.g. Slurm [7]) and the InfiniBand's subnet manager (OpenSM). If Slurm can forward the job-to-node mapping to OpenSM, then the static routing tables can be recalculated on a regular basis to increase the utilization of network resources within a job and among multiple jobs.

2.1 Filtering Multi-Switch Batch Jobs

We choose a loose coupling of Slurm and OpenSM via an additional application, which periodically queries Slurm for running jobs. It analyzes the job-to-node mapping to find parallel applications which use more than one IB switch, and therefore could benefit from an optimized path balancing. If the queried set of these multi-switch jobs differs from a previous configuration, then our tool stores the new job-to-node mapping and sends a SIGHUP to OpenSM enforcing a re-routing of the fabric.

2.2 Job-Aware DFSSSP Routing

The approach we take is to use DFSSSP routing [3] as the basis for our job-aware routing, since it is known to offer high path balancing for regular and irregular network topologies. Furthermore, it allows us to reuse other existing capabilities of DFSSSP, such as minimal path lengths, separate optimization for MPI and I/O traffic, and avoidance of credit loops [2].

Three extensions for DFSSSP are implemented to enable job-aware routing. First, the list of job IDs running on each compute node, see Section 2.1, is added to the internal port/node representation. Second, the processing of ports has to be sorted (clustered by jobs and decreasing in job size) to meaningfully make use of DFSSSP balancing feature. And last, the adjustment of link weights within DFSSSP has to distinguish between inter- and intra-job paths.

3. EVALUATION FOR HPC WORKLOADS

We evaluate our JA-DFSSSP routing based on two routing/network metrics. First, the edge forwarding index (EFI)

of network ports for each application, i.e., the number of intra-application paths per port. A decrease in maximum EFI theoretically reduces the congestion within an application or among competing applications. The second metric is the number of inter-switch links (ISL) used by an application or set of applications. Increasing the ISL indicates an increased fabric utilization and an increased throughput available to each application.

The evaluation is performed for the 509-node HPC system *Taurus* located at the ZIH, TU Dresden. Taurus' 494 compute nodes are distributed over three IB islands and connected via 52 36-port FDR IB switches. Each island utilizes a full bisection bandwidth fat-tree topology.

3.1 Artificial Workload

We use one island of Taurus for an initial comparison of DFSSSP routing to our job-aware version. Three 60-node jobs are distributed in interleaving chunks of 30 nodes across the 180 compute nodes, in the order: $j_1, j_2, j_1, j_3, j_2, j_3$, where job 1 uses nodes 1-30, job 2 uses nodes 31-60, and so forth. The analysis of the network metrics reveals that the maximum EFI decreased by 59% per job while the ISL increased by 7% in total.

3.2 Realistic HPC Workload

For a more in-depth analysis we use a realistic HPC workload: Taurus' batch job history of a complete month (February 2015) and compare our JA-DFSSSP routing to DFSSSP and Up*/Down* [9] (Taurus uses the latter).

4. RELATED WORK

Communication performance optimization for multi-job HPC environments has been studied in the past resulting in different approaches. Mellanox's traffic-aware routing, TARA [10], balances paths based on link/port usage. This is a reactive optimization without knowledge of job locality. Application-aware routings [8] have been developed, but require detailed knowledge about traffic patterns and injection rates of each application. Optimizing job-to-node assignment via the batch system, while considering the network topology [5, 12], is feasible for regular networks, but assumes idealized routing or ignores it completely. Performing routing-aware job-to-node mapping [1] by the job scheduler is another option. However, the computational complexity to solve the mapping problem is infeasible for an online scheduling of jobs.

5. CONCLUSION

We showed that our job-aware routing extension of DFSSSP is a viable alternative to the state-of-the-art routing algorithms available for InfiniBand networks. JA-DFSSSP offers a lower edge forwarding index on a "per job" basis in HPC-system used simultaneously by multiple applications and users, and is able to increase the fabric utilization for realistic HPC workloads. Considering the job locality, rather than the actual traffic, allows for a preemptive path optimization. Our approach of using an interface between the batch system and routing engine does not rely on a particular batch system and it's scheduling algorithm, and is therefore more universally usable than extending scheduling algorithms to be routing-aware and/or topology-aware.

6. REFERENCES

[1] A. H. Abdel-Gawad, M. Thottethodi, and A. Bhatele. RAHTM: Routing Algorithm Aware Hierarchical Task Mapping. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '14, pages 325–335, Piscataway, NJ, USA, 2014. IEEE Press.

[2] W. J. Dally and C. L. Seitz. Deadlock-Free Message Routing in Multiprocessor Interconnection Networks. *IEEE Trans. Comput.*, 36(5):547–553, 1987.

[3] J. Domke, T. Hoefler, and W. E. Nagel. Deadlock-Free Oblivious Routing for Arbitrary Topologies. In *Proceedings of the 25th IEEE International Parallel & Distributed Processing Symposium (IPDPS)*, Washington, DC, USA, 2011. IEEE Computer Society.

[4] T. Hoefler, T. Schneider, and A. Lumsdaine. Multistage Switches are not Crossbars: Effects of Static Routing in High-Performance Networks. In *Proceedings of the 2008 IEEE International Conference on Cluster Computing*. IEEE Computer Society Press, 2008.

[5] T. Hoefler and M. Snir. Generic Topology Mapping Strategies for Large-scale Parallel Architectures. In *Proceedings of the International Conference on Supercomputing*, ICS '11, pages 75–84, New York, NY, USA, 2011. ACM.

[6] InfiniBand Trade Association. *Infiniband Architecture Specification Volume 1, Release 1.2.1*, 2007.

[7] M. A. Jette, A. B. Yoo, and M. Grondona. SLURM: Simple Linux Utility for Resource Management. In *In Lecture Notes in Computer Science: Proceedings of Job Scheduling Strategies for Parallel Processing (JSSPP) 2003*, pages 44–60. Springer-Verlag, 2002.

[8] M. A. Kinsy, M. H. Cho, T. Wen, E. Suh, M. van Dijk, and S. Devadas. Application-aware Deadlock-free Oblivious Routing. In *Proceedings of the 36th Annual International Symposium on Computer Architecture*, ISCA '09, pages 208–219, New York, NY, USA, 2009. ACM.

[9] Mellanox Technologies. *Mellanox OFED for Linux User Manual*, rev 2.0-3.0.0 edition, 2013.

[10] Mellanox Technologies. Product Brief: Unified Fabric Manager Software. http://www.mellanox.com, 2015. [Online; accessed 07-August-2015].

[11] TOP500. http://www.top500.org, 2015. [Online; accessed 07-August-2015].

[12] H. Yu, I.-H. Chung, and J. Moreira. Topology Mapping for Blue Gene/L Supercomputer. In *Proceedings of the ACM/IEEE SC 2006 Conference*, pages 52–52, Nov 2006.