Increasing Fabric Utilization with Job-Aware Routing

Improvement of Network Metrics for a Multi-Job HPC Environment

Our job-aware DFSSSP (JA-DFSSSP) routing is designed to increase the IB fabric utilization of an HPC-system, which is used by many users for varying workloads. The figure to the right compares our JA-DFSSSP to the following two IB routing algorithms:

a) Up*/Down* routing, a common choice in production HPC-systems (incl. Taurus)
b) the “default” DFSSSP-routing, as implemented in OFED’s OpenSM v3.3.18

For the analysis we used the March 2015’s workload (job size/locality) of the 509-node HPC-system Taurus, located at TU Dresden. We compare edge forwarding index (EFI), and the number of inter-switch links (ISL) used on a “per job” basis and for the sum of jobs running concurrently. Analyzing the effects of different routings on realistic HPC workloads reveals:

a) DFSSSP vs. UpDn: maximum EFI by a value of 8 and ISL by 66 on avg. per job
b) JA-DFSSSP vs. DFSSSP: max. EFI by a value of 50 and ISL by 64 on avg. per job
c) avg. boost of ISL by JA-DFSSSP is 10% (6%) considering all concurrently running jobs compared to DFSSSP (and Up/Down)
d) max. ISL / EFI was 19% (16%/21%)

This indicates a noticeable communication performance improvement thru JA-DFSSSP.

Introduction

The InfiniBand fabric of an HPC-system is a shared resource for simultaneously running applications. The communication performance of these scientific codes depends heavily on the used routing algorithm. Many of the oblivious routing algorithms in the InfiniBand subnet manager [1] optimize for global path balancing which is suboptimal in a multi-job environment. An alternative approach is to attribute as much physical network components (links, switches) as possible to each multi-node application. This can be accomplished by our job-aware DFSSSP [2] routing and Slurm extension [3].

Multi-User/Multi-Job HPC System

Large HPC-systems at national labs and universities are usually used by many users running a diverse set of serial and parallel applications (differ in node count, runtime, and communication pattern, etc). Our intended optimizations are only relevant to multi-node jobs using more than one IB switch in the fabric. We analyze the job history of the Taurus HPC-system (494 compute + 15 I/O nodes in three “full” fat-free islands; using 52 36-port FDR IB switches), see statistics of March 2015 on the right. Two facts are surprising:

a) only 66.8% (avg.) of nodes are used by multi-switch jobs
b) a large number of small jobs (size range: 2-18 nodes) is unnecessarily distributed over multiple switches.

Interface between Slurm and OpenSM

We designed a program which analyzes the size and location of running jobs and triggers a rerouting of the fabric (via OpenSM) to optimize the path balancing. This program will identify all running multi-node jobs which have their compute nodes connected to more than one switch. The detailed operational steps are:

while (TRUE):
  query queue for all running jobs;
  get topology (node-switch map);
  filter multi-node, multi-switch jobs;
  if (job-to-node mapping changed):
    write new job-to-node to disk;
    send SIGHUP to OpenSM;
    sleep defined minutes;

The verification if the job-to-node mapping has changed compared to previous iterations ignores the actual job IDs, since a new job (same size/location) does not require recomputation of the forwarding tables.