D-Mod-K Routing Providing Non-Blocking Traffic for Shift Permutations on Real Life Fat Trees

Eitan Zahavi

***Abstract*—As the size of High Performance Computing clusters grows, the increasing probability of interconnect hot spots degrades the latency and effective bandwidth experienced by MPI collectives. This report presents a proof that using D-Mod-K routing scheme for real life constant bisectional-bandwidth fat-tree topologies solves this problem for Shift Permutations traffic patterns.**

***Index Terms*—Network Topologies, Routing Algorithms and Techniques**

I. Introduction

In recent years, the ever increasing demand for compute power is addressed by building multiple thousand nodes High Performance Computing (HPC) clusters exceeding the peta-flop per second barrier [1]. State of the art parallel programs running on these clusters utilize the standard Message Passing Interface (MPI). Message passing overcomes the need to implement concurrent shared memory among the distributed processes at the cluster nodes. Many MPI applications exhibit a ratio of communication to computation time which is proportional to the number of nodes they use. For these parallel applications, the network latency and bandwidth may inhibit the desired linear performance acceleration with cluster size. There are two main network topologies used by HPC clusters: fat-trees and 3D tori. Fat-trees are commonly used for variable and diverse traffic patterns.

With the rise in HPC cluster size, a recent study [2] has measured degradations of network performance, down to 40% of the nominal bandwidth provided by fat-trees to MPI communication. The source of the degradation is attributed to cases where traffic from multiple sources congests a particular network link, creating a "hot spot". Our simulations reproduced these results, predicting the average degradation is 40% of the nominal network bandwidth for random job assignment. Furthermore, for adversarial job assignment, degradation to 7.1% of the network bandwidth was simulated.

Manuscript received Aug 16, 2010. This work was supported in part by Mellanox Technologies LTD.

Eitan Zahavi is with the Electrical Engineering Department, Technion, Haifa, 32000 Israel and with Mellanox Technologies, Yokneam, 20692 Israel (phone: +972-544-478803; e-mail: ezahavi@tx.technion.ac.il).

Traditionally, it was assumed that the primary bottleneck for performance scalability in HPC is the operating-system latency jitter. However, with the availability of and clock synchronization protocols and Collectives Offloading solutions to the jitter, network hot spots has recently become the new limiting factor for system growth [19].

This report holds a proof that a d-mod-k routing solution for a class of practical fat-tree topologies is able to route shift permutation traffic with no hot-spots.

The report is organized as follows: Background for fat-tree topologies is provided in section II. Section III describes the problem of network scalability. The D-Mod-K routing is presented in section IV and the proof in section V.

II. Real Life Fat-Tree Formulation

This section forms the basis for the discussion of non blocking routing in fat-trees. Although there are several fat-tree representations in the literature, there is no existing formulation that is sufficient to describe real life fat-trees used within today's HPC clusters. The progress from k-ary-n-trees [11][12] through Extended Generalized Fat-Trees [13], to this paper contribution of Parallel Ports Fat-Trees (PGFTs) and their sub-classification into Real Life Fat-Trees (RLFTs) will be discussed in the following. The notations developed in this section are used later in the proof of the proposed routing non-blocking properties (presented in the appendix).

A. Why Parallel Ports Generalized Fat-Trees are required?

The term fat-tree stems from the idea that if a single rooted tree of cross-bar-switches could be built (each node with K connections down and one connection up towards the root of the tree), in order to preserve bandwidth, its links towards the root of the tree should be K times "fatter" then it's down links. Such a tree of height h requires the top link to have a bandwidth which is larger by a factor of K^h than the bandwidth of a leaf link. This requirement renders such trees impractical as different speeds are required from switches and links at different levels in the tree. To overcome this requirement, the concept of a multi-rooted topology was introduced. However, in multi-rooted trees the non-blocking nature of the single rooted tree is replaced by a weaker attribute: they are rearrangeably-non-blocking i.e. given a permutation of source and destination pairs the routing on the tree can be made non-blocking.