MPI aware routing using SDN

Siddharth Bhatia, Yash Sinha, G S S Chalapathi, Rajiv Kumar

BITS, Pilani.

ABSTRACT

Among the factors that determine the performance of a computationally intensive application running on a high-performance computing (HPC) system, communication between processes is vital. Our fundamental idea behind optimizing Message Passing Interface (MPI) communications is to maximize the utilization of the network of the cluster system, by deploying the Software Defined Networking (SDN) paradigm. We identify two primary issues: overuse of shortest paths leaving them choked and dynamically un-optimized path selection. SDN will be used to leverage dynamic routing to avoid these two issues. In this paper, we present an application-aware network routing mechanism specifically for enhancing MPI applications with the help of an adaptive routing algorithm.

INTRODUCTION

High performance computing (HPC) systems enable users to solve large scale, complex and computationally expensive problems

Message Passing Interface (MPI) has been the standard specification for message passing programming required in computing programs that run on cluster systems. It has been concluded that most of the applications largely use collective communication during their execution. Therefore, it is crucial to reduce the time spent in collective communication which is directly proportional to performance.

To manage communication between multiple processes, proper manageability and central control of the network is required which has been boldly proposed in the new network paradigm viz., Software Defined Networking (SDN). It separates the control plane from the data plane (which are traditionally coupled together) and introduces a centralized controller which pushes proper routing entries in the forwarding elements (erstwhile switches). Further, the controller exposes an interface to interact with other applications (such as an MPI program) that can be used to pull network statistics and push network control. Various researchers have tailored the use of SDN to enhance different network applications. Takahashi et al. present a generic SDN enhanced MPI framework.

We extend this concept to formulate a design of MPI aware dynamic routing with the help of SDN controller. We wish to enhance MPI based HPC systems by leveraging SDN in network routing control to address MPI communication over choked links and dynamically un-optimized paths by optimizing path selection with dynamic network monitoring from the controller as elaborated in the next section. Further, we delineate the use of a specific adaptive routing algorithm in the context of SDN.

NETWORK ISSUES IN MPI SYSTEMS

MPI collective operations require much simultaneous communication among multiple process pairs. However, while some computing node pairs communicate less, some pairs have to communicate much more than other pairs. When the underlying network of the cluster system interconnects with the computing nodes there is a possibility of link contention, redundant routing, un-optimized path selection, and many more issues. The communication in MPI uses the network links in various ways but we focus on the following:

Using choked links:

Suppose we have a cluster system of 4 computing nodes interconnected with a 2-level Fat-tree topology as shown in Figure 1. Fat-tree is a network topology that has multiple redundant routes between the upper layer switches and lower layer switches. By scattering traffic among these available routes, it is possible to gain more bandwidth compared with simple tree topology. Exploiting multiple available paths has been investigated and used widely, but with SDN, design of a new protocol such as MPTCP is not required.

Therefore, for such a setup, the load balancing algorithm becomes important. If two communications are routed within the exact same route, link contention could happen. On the other hand, when two communications are re-routed so that they make use of the available routes, link contention can be avoided. Traditionally, this has been achieved with modification of IS-IS/OSPF link weights, but this is viewed ``as a significant change to the network that is performed on a relatively coarse timescale''. Hence, this is not suitable for frequent changes, which on the contrary lead to worse situation of flow oscillations.

SDN can easily overcome this with frequent rule installations at the switches with the help of the controller.

Using un-optimized paths:

Using the same topology as above, the nodes may end up routing packets on un-optimized paths, especially if heuristics are being used to avoid contention and path selection without knowing the global view of the network in real time. In practice, the traffic volumes do not remain static but fluctuate over time, and unexpected failures can result in changes to the network topology. In addition, acquiring an exact estimate of the traffic matrix is difficult.

With SDN we can retrieve the network state statistics with help of statistics request messages and accordingly provision and re-provision the paths that not only suit the static parameters like path length and capacity but also dynamic parameters like available bandwidth, link delay etc.

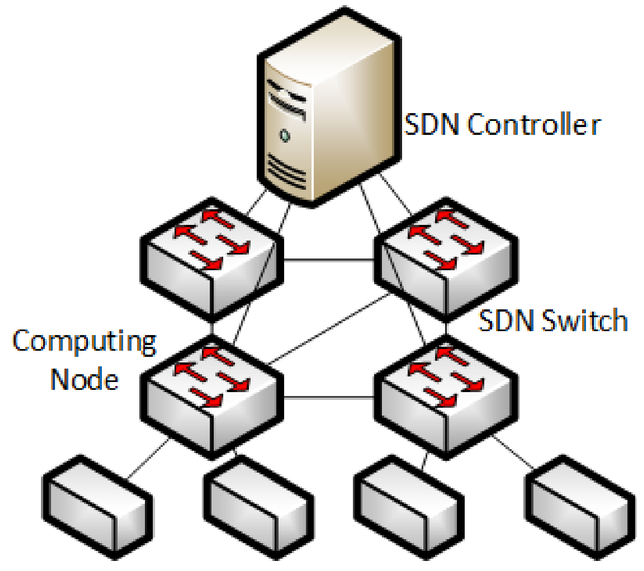

Figure 1. MPI Cluster with SDN

METHODOLOGY

The controller is a single computer that configures the network in an MPI application-aware manner. As shown in Figure 2, all the switches are connected with the controller. They are merely responsible for forwarding the packets based on the routes dictated by the controller whereas, the controller implements all the network control logic.

The controller maintains the global topology of the network as an abstract network graph with link weights between the nodes. The controller polls for flow statistics from the switches periodically and estimates link bandwidth and delay to update the link weights in the graph. We use Self-Adaptive Multiple Constraints Routing Algorithm (SAMCRA) algorithm. At each hop along the path, SAMCRA computes the next hop towards the destination, ignoring the previous path history (as in current IP routing).

At the source switch, the controller determines the path for the flow based on current weights and installs rules in the subsequent switches. The controller keeps an eye on the congestion prone areas, and re-provisions paths for flows to avoid congestion. This can enhance all communications viz., point-to-point and collective in MPI.