CMSC416: Introduction to Parallel Computing

Topic: Parallel Networks and Filesystems
Date: April 25th, 2024

Routing Algorithms

A Routing Algorithm decides how a packet is routed between source and destination switch.

Static Routing is a preprogrammed route designated by a routing table.

Dynamic Routing is where routing changes at runtime. In this routing the network can adapt for a myriad of reasons, not just congestion.

Adaptive Routing is a modification of dynamic routing that adapts to network congestion.

Network Congestion

Performance variability in programs is due to congestion derived from placement of jobs and contention for network resources. The factors that contribute to the variability are both the **placement of jobs** and **contention for network resources**.

Torus networks are particularly susceptible to congestion issues as data shows that nodes with many neighboring node jobs tend to have higher times, as there is communication running through them.

Mitigating Congestion

Network congestion can be mitigated by Network Topology Aware Node Allocation, Adaptive Flow Aware Routing, and Topology Aware Mapping.

Network Topology Aware Node Allocation

Based on the topology of your network, there are multiple heuristics you can use to reduce congestion. Factory networks have greater chances of congestion as communication between nodes on different pods or switches needs to go up the tree. By taking into consideration this topology it is important to **place jobs on the same**

switch or pod when you know they need to communicate. As a general rule you should allocate nodes to prevent the sharing of links by multiple jobs.

Adaptive Flow Aware Routing

In this method the network will dynamically reroute traffic to alleviate hot spots based on the current load for each pair of nodes.

Steps:

1. Calculate the current load on all links
2. Find the link with maximum load
3. If maximum exceeds an arbitrarily set threshold, then reroute one flow to an under utilized link.

Important to note that in general this process only needs to run once for every time a new job is added as that would dictate a change in traffic. For more complex jobs that may contain internal load balancing this may not be the case.

Topology Aware Mapping

In this method you need inputs of the application communication graph and the machine topology. Using these inputs within a job allocation you map processes to nodes intelligently for better communication performance.

Parallel Programs Input/Output

Parallel programs utilize input/output operations to read input datasets, write numerical output, and write checkpoints.

Non Parallel I/O

Where one designated process performs I/O and all processes send/receive data from that process.

Parallel File System(I/O sub-system)

In a parallel file system, home directories and scratch space are mounted onto the login and compute nodes. The system consists of the metadata server/target and the object storage server/target.

The **Metadata Server(MDS)** handles requests from compute/login nodes for information about file metadata such as permission, ownership, creation, and directory structure.

The **Metadata Target(MDT)** is the hardware that stores the metadata itself that the server will retrieve from.

The **Object Storage Server(OBS)** manages the actual file data stored in the parallel file system and like the MetaData Server handles requests from the nodes. The compute nodes are only virtually mounted and the actual data is accessed through the OSS.

The **Object Storage Target(OBT)** is the hardware that stores the actual file data that the server interfaces with.

The compute nodes communicate with the OSS through an underlying daemon that checks what the compute node is accessing and gives it. The packets of information begin from the compute node and then they go to the LNET router node which accesses the leaf switch of the OSS. The OSS then performs the I/O operation and handles any requests.

Parallel file systems offer many benefits for both performance and structure. The parallel file systems have improved I/O bandwidth by spreading read/writes across OSTs. They also allow for **striping** where you can have one file across multiple OSSs.

High Performance Computer systems have long term storage called **tape drives**. These tape drives are data stoered on magnetic tapes mostly for archiving purposes. The data takes a long time to be copied over to parallel file systems for use.

Burst Buffer is a fast intermediate storage between the filesystem and compute nodes. The storage is slower than the DRAM but has a higher capacity. Likewise the storage has a lower capacity than disk storage but a faster speed. If you are aware that you will need some data soon you can bring it into the burst buffer to reduce latency later on. Can be used to store checkpoint data and prefetch input data.