CMSC416 Notes - Thursday, April 25, 2024

1. Routing Algorithm
    a. Decides how a packet is routed between a source and destination switch
    b. Static routing:
        i. Factory Network Example
        ii. Each router is pre-programmed with a routing table
            1. Decided at boot time
    c. Dynamic routing:
        i. Can change the routing at runtime
        ii. While the machine is running, you can change for pairs of routers
        iii. Done by software
    d. Adaptive routing
        i. A subset of dynamic routing, which also incorporates the network's congestion into making dynamic routing changes
2. Performance Variability
    a. Operating System Noise/Jitter
    b. Network Congestion
3. Performance Variability caused by Network Congestion
    a. No variability in computation time, all the time change is driven by communication performance
    b. Determined by:
        i. Job placement
        ii. Network Resource Demand
4. Mitigating Congestions
    a. Define convex shapes to jobs → This was the IBM approach with (8x8x8) chunks
    b. Adaptive Routing
    c. Topology Aware mapping of existing processes
        i. "Can I map my job better"
5. Topology Aware Node Allocation
    a. Want to define heuristics to minimize travel distance
    b. The higher you go on the nodes, the more likely you are to interact with traffic from other jobs
    c. One possible heuristic is to schedule all processes on one network switch.
    d. Another heuristic is, if you have a number of nodes that doesn't fit on one switch, schedule them next to each other, in the same pod
    e. In general, allocate nodes in a manner that prevents sharing of links by multiple jobs while maintaining high utilization
6. Adaptive Flow Aware Routing (AFAR)
    a. Given traffic for each pair of nodes in the system and the current routing
        i. Calculate current load on each link
        ii. Find the link with the max load

        iii. If max > threshold, re-reroute one flow crossing that link to an under-utilized link
        iv. Repeat
    b. This process eliminates hotspots!
7. Input/Output Operations
    a. Reading data
    b. Writing checkpoints or numerical outputs
8. Non-Parallel IO
    a. In class, we typically set one process doing IO, and the other processes send and recv info from one process
    b. However, this is not scalable
9. Parallel File System
    a. Home dirs and scratch are typically on a parallel file system
    b. Mounted on login and compute nodes
    c. Cluster and FS are connected via links
    d. OSS (object storage server) nodes are connected via network → separate topology
    e. I/O nodes have their own cluster, compute nodes have their own cluster, connected by intermediate
10. Parallel File System Improvements
    a. Improves I/O bandwidth by spreading Reads and writes
    b. Each compute node runs an IO daemon to interact with its filesystem
11. Tape drive
    a. Storage data on magnetic tapes, typically an archival step
12. Burst Buffer
    a. Fast, intermediate storage between compute nodes and the parallel FS
    b. Two Designs
        i. Node Local
        ii. Remote
    c. Can be used to store checkpoint data, fetching input data ahead of time, workflows that need to be analyzed as they happen
13. I/O Libraries
    a. HDF5 and NetCDF
        i. Hierarchical format for n-dimensional data
    b. Middleware: MPI-IO
        i. POSIX support
    c. POSIX IO
        i. Standard Linux/Unix
14. I/O Patterns
    a. One process reading/writing
    b. Multiple processes reading and writing to a shared file
    c. Multiple processes reading and writing to and from different files
    d. Performance depends upon number of readers and writers (based on threads)
15. I/O profliers
    a. Darshan → Argonne NL

b. Recorder → UIUC research tool, tracing style tool, provides support for different I/O libraries as mentioned above