Large Scale Data Processing CSE3025

Dr. Ramesh Ragala

March 18, 2021

File Management System

- Two main purposes of using files:
  - Permanent storage of information on a secondary storage media
  - Sharing of information between applications
- A file system is a subsystem of the operating system that performs file management activities such as organization, storing, retrieval, naming, sharing, and protection of files.

Need of Distributed File System

- The only feasible approach to tackling large-data problems is to divide and conquer (it is a fundamental problem in Computer Scicence)
- Idea of Divide and Conquer approach: Partition a large problem into smaller sub-problems $\rightarrow$ these sub-problems are independent $\rightarrow$ they can be tackled in parallel by different compute nodes (threads in a core processor, cores in multi-processor, multiple processors in a machin, many machines in a cluster).
- Intermediate results from each individual worker are then combined to yield the final output
- However, the details of their implementations are varied and complex.
    - How do we break up a large problem into smaller tasks? More specifically, how do we decompose the problem so that the smaller tasks can be executed in parallel?
    - How do we assign tasks to workers distributed across a potentially large number of machines
    - How do we ensure that the workers get the data they need?
    - How do we coordinate synchronization among the different workers?

Distributed File System

- Mostly computing is done on a single processor, with its main memory, cache, and local disk $\rightarrow$ compute node.
- Applications that called for Parallel processing such as large scientific calculations were executing on special-purpose parallel computers with many processors and specialized hardware.
- The prevalence of large-scale Web services has caused more and more computing to be done on thousands of computing nodes operating more or less independently.
- Moore's law suited $\rightarrow$ building bigger and bigger servers is no longer necessarily the best solution to large-scale problems. $\rightarrow$ An alternative solution that has gained popularity is to tie together many low-end/commodity machines together as a single functional distributed system.
- Distributed system $\rightarrow$ Scale-Out
- Goal of DFS: provide common view of centralized file system, but distributed implementation.

Distributed File System

- In addition to the functions of the file system of a single-processor system, the distributed file system supports the following:
  - Remote information sharing
  - User mobility → User should be permitted to work on different nodes.
  - Availability
  - Diskless workstations → DFS, with its transparent remote-file accessing capability, allows the use of diskless workstations in a system.
- Desirable features of a distributed file system:
  - Transparency → Structure transparency, Access transparency, Naming transparency, Replication transparency, etc.
  - User mobility
  - Performance
  - Simplicity and ease of use
  - Scalability
  - High availability
  - High reliability
  - Data integrity
  - Security
  - Heterogeneity

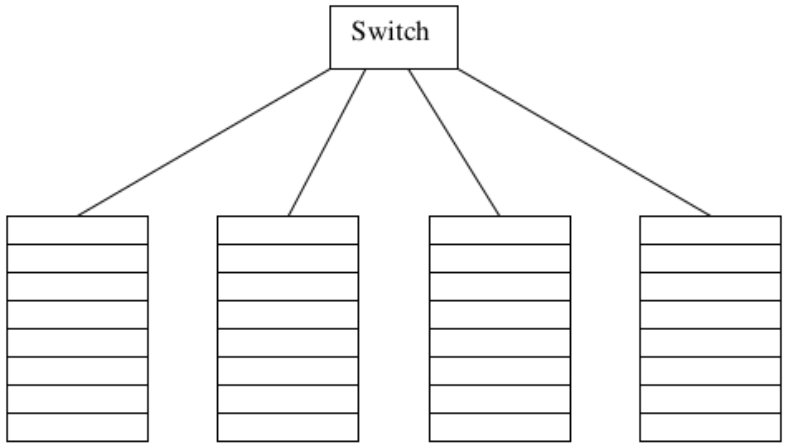
Simple Scenario

A high-end machine with four I/O channels each having a throughput of 100 MB/sec will require three hours to read a 4 TB data set! With Hadoop, this same data set will be divided into smaller (typically 64 MB) blocks that are spread among many machines in the cluster via the Hadoop Distributed File System (HDFS). With a modest degree of replication, the cluster machines can read the data set in parallel and provide a much higher throughput. And such a cluster of commodity machines turns out to be cheaper than one high-end server

Distributed File System

- The compute nodes are commodity hardware, which greatly reduces the cost compared with special-purpose parallel machines.
- New computing facilities have given support to a new generation of programming systems. $\rightarrow$ the power of parallelism.
- At the same time avoid the reliability problems that arise when the computing hardware consists of thousands of independent components, any of which could fail at any time.
- Design of specialized file system that have been developed to take these advantages

- **Physical Organization of Computing Nodes:**
  - A New parallel-computing architecture → sometimes called as "Cluster Computing".
  - Compute nodes are stored on racks, perhaps 8 - 64 on a rack.
  - The nodes on a single rack are connected by a network → Gigabit Ethernet
  - There can be many racks of compute nodes, and racks are connected by another level of network or a switch.
  - The bandwidth of inter-rack communication is somewhat greater than the intrarack Ethernet,



Racks of compute nodes

- **Problems:**
  - Failure of Computing nodes $\rightarrow$ loss of single node
  - Failure of Interconnection networks $\rightarrow$ loss of entire rack
- Difficult to restart or abort the computation for every component failure. What happens if any one of the applications takes more time to finish during this kind of situations. $\rightarrow$ The applicaiton may not be complete successfully.
- Solutions to this problem:
  - Files must be stored redundantly
    - No duplicate of the file at several compute nodes $\rightarrow$ then if one node failed, all its files would be unavailable until the node is replaced.
    - If we did not back up the files at all, and the disk crashes, the files would be lost forever.
  - Computation must be divided into tasks.
    - In this case, if any one task fails to execute to completion, it can be restarted without affecting other tasks.
    - MapReduce uses this prinicple.

Hadoop -Introduction

- An Open source software framework (Apache Project)
- In this Framework, users can write and run the distributed applications that process massive dataset.
- what makes it especially useful
  - Scalable: It can reliably store and process petabytes.
  - Economical: It distributes the data and processing across clusters of commonly available computers (in thousands).
  - Efficient: By distributing the data, it can process it in parallel on the nodes where the data is located.
  - Robust and Reliable: Hadoop is architected with the assumption of frequent hardware malfunctions. It automatically maintains multiple copies of data and automatically redeploys computing tasks based on failures.
  - Simple and Accessible: It runs on large clusters of commodity machines or on cloud computing services