GIGAOM RESEARCH

Apache Hadoop: Is one cluster enough?

Paul Miller

December 15, 2014

This report is underwritten by WANdisco.

Table of Contents

Table of Contents 2

Executive summary 3

Hadoop at work 4

YARN as enabler 5

HDFS 6

Moving beyond a single cluster 7

Next steps 12

Key Takeaways 13

About Paul Miller 14

About Gigaom Research 14

Executive summary

The open-source Apache Hadoop project continues its rapid evolution now and is capable of far more than its traditional use case of running a single MapReduce job on a single large volume of data. Projects like Apache YARN are expanding the types of workloads for which Hadoop is a viable and compelling solution, which is leading practitioners to think more creatively about the ways data is stored, processed, and made available for analysis.

Enthusiasm is growing in some quarters for the concept of a "data lake" — a single repository of data stored in the Hadoop Distributed File System (HDFS) and accessed by a number of applications for different purposes. Most of the prominent Hadoop vendors provide persuasive examples of this model at work but, unsurprisingly, the complexities of real-world deployment do not always neatly fit the idealized model of a single (huge) cluster working with a single (huge) data lake.

In this report we discuss some of the circumstances in which more complex requirements may exist, and explore a set of solutions emerging to address them.

Key findings from this report include:

- YARN has been important in extending the range of suitable use cases for Hadoop.
- Although mainstream Hadoop deployments still largely favor a single cluster, that simple model does not make sense for a range of technical, practical, and regulatory situations.
- In these cases, deploying a number of independent clusters is more appealing, but this fragments the data lake and risks reducing the value of the whole approach. Techniques are now emerging to address this challenge by virtually recreating a seamless view across data stored in different physical locations.

Hadoop at work

Hadoop has come a long way since 2005, when Doug Cutting and teams at Yahoo and elsewhere created a working implementation of Google's MapReduce ideas. Even in early versions of the model, Hadoop's MapReduce capabilities made it possible for teams with a set of relatively cheap commodity servers and a degree of technical ability to analyze data of a volume and complexity that would previously have required extremely expensive proprietary hardware and software.

With Hadoop, commodity off-the-shelf servers could be grouped together in clusters to analyze very large data sets. MapReduce lay at the heart of Hadoop and was the engine that powered the process of chopping data sets into manageable chunks, analyzing each of these separately and in parallel, and then re-combining the results. The rest of Hadoop, initially, was simply concerned with the logistics of managing the cluster: sharing tasks between available machines or recovering from the failure of individual machines, for instance.

MapReduce was — and remains — a powerful tool for the batch-processing of large static data sets, but it is less-suited to other data analytics workloads, including interactive data analysis, streaming data from sensors and social networks, or complex in-memory processing. As interest in and use of Hadoop grew, there were frequent calls to extend its capabilities beyond batch processing. A number of projects and products supported this move, but Hadoop's core dependence upon MapReduce created a performance bottleneck that was difficult to work around.

YARN as an enabler

Introduced with version 2 of Apache Hadoop, "Yet Another Resource Negotiator" (YARN) is a key component in the Hadoop community's effort to broaden the project's reach. YARN takes on the role of managing resources in a Hadoop cluster, removing the project's core dependence upon MapReduce and making it possible for non-MapReduce workloads to share the cluster's resources. Hadoop's MapReduce module has been rewritten, removing its cluster management capabilities and focusing exclusively upon the specifics of running MapReduce jobs. The underlying APIs maintain compatibility with earlier versions of Hadoop.

YARN is designed to deliver a flexible resource-management solution capable of supporting multiple applications running on a single Hadoop cluster simultaneously. MapReduce jobs, stream processing in Storm, and graph-data analysis in Giraph could, in principle, all run simultaneously on a single cluster, with YARN managing the allocation and reallocation of resources as demand shifts. Cluster resources are better managed than in earlier versions of Hadoop, and companies depending on Hadoop for multiple use cases should realize benefits from more cost-effective utilization of their investment.

YARN is widely regarded as key to Hadoop's continued diversification and growth, but real industry support for its capabilities remains incomplete. Hortonworks, a prominent supporter of the Apache project that became YARN, remains perhaps the most vocal advocate for YARN and pushes it within the company's own distribution of Hadoop. Competing Hadoop distributions from Cloudera, MapR, Pivotal, and others also now include YARN, but it will take time for all of the projects and products in the Hadoop ecosystem to fully realize the advantages of natively implementing YARN's way of working.

The Hadoop Distributed File System

YARN plays an important and growing role in managing data processing tasks across all of the compute resources gathered together within a Hadoop cluster. Behind all of the MapReduce jobs, stream analyses and other data processes that a Hadoop cluster typically performs sits another shared component of the Hadoop architecture which is just as important: the Java-based Hadoop Distributed File System (HDFS).

HDFS is Hadoop's distributed and scalable file system, designed to store all of the data required by a Hadoop cluster and capable of scaling so that it can potentially store hundreds of petabytes of data across thousands of nodes. Recognizing the possibility of hardware failures in clusters that include large numbers of commodity servers, HDFS is designed to be reliable and fault-tolerant. Data is typically stored on at least three nodes within the cluster, reducing the risk of data loss. The Hadoop cluster maintains an awareness of where data is stored, enabling it to optimize the allocation of tasks to different nodes within the cluster. Specific tasks will typically be allocated to idle compute nodes located near the data to be analyzed, reducing the need for data transfer over the network and significantly improving performance within the cluster. A NameNode is responsible for storing metadata about the data within an HDFS cluster, tracking where data is stored, ensuring that data is replicated and that those replicas are maintained. Individual DataNodes within the cluster store the data they are allocated, and make it available to Hadoop processes when required.

The data lake

Combined, YARN and HDFS deliver a powerful set of capabilities on which a wide and growing range of Hadoop data processing tasks may draw. HDFS's ability to store large volumes of data reliably and redundantly along with YARN's growing value in managing competing demands for cluster resources underpin industry enthusiasm for the idea of a data lake.

Taken to its logical extreme, the data lake is a single source for all the data relevant to a single project, workflow or even company. Instead of individual applications storing their own data — and often duplicating data held elsewhere — the data lake model proposes that multiple applications should all contribute to and draw from a single, shared set of data. With HDFS storing data and YARN mediating access by different applications running within a single cluster, Hadoop vendors increasingly suggest that they now have the means for delivering an effective and useful data lake. Companies like Pivotal investor GE are beginning to realize some of the value that the approach offers, enabling connections to be made in previously separate data sets and reducing the time required to achieve results.

Moving beyond a single cluster

Before YARN was introduced, scheduling Hadoop workloads on a cluster was a fairly primitive process with limited ability to prioritize tasks and an inefficient dependence upon MapReduce. With YARN, sharing a single Hadoop cluster's resources between several jobs all running at the same time is relatively straightforward. This capability of YARN makes it far easier to provision and manage a single Hadoop cluster and, particularly in environments where use is variable and bursty in nature, typically leads to significant cost savings.

Despite the clear benefits introduced by YARN, a number of situations still exist in which maintaining separate Hadoop clusters makes sense. These include:

- **Workload optimization**. Hadoop now powers a wide and growing set of workloads, including batch analysis with MapReduce, SQL-like interaction with structured data using tools like Hive or Impala, NoSQL transactional systems based upon HBase or Accumulo, in-memory analytics powered by Spark, or stream processing with Storm. These workloads all raise slightly different considerations in terms of bandwidth requirements, processor utilization, memory usage, storage capacity, or the speed and frequency of data reads and writes. Although YARN is responsible for allocating workloads to available cluster resources, it mostly assumes that nodes within the cluster are essentially equivalent to one another. It is not currently good at consistently recognizing and exploiting the different capabilities of individual nodes within the cluster. While it is certainly feasible to run some or all of these different workloads on a single Hadoop cluster, there may be value in continuing to maintain additional clusters in which configurations are highly optimized to the specific requirements of particular — valuable or mission-critical — workloads.

- **High cluster utilization**. In situations where an existing cluster is routinely showing very high levels of utilization dominated by two or more different workloads, considering either adding additional resources to the cluster or creating separate clusters with more managed usage patterns makes sense.

- **Separating work from play**. Hadoop is increasingly being used for real and mission-critical workloads within a wide range of organizations. At the same time, developers inside those organizations continue exploring new opportunities, testing new components, and pushing the boundaries of what they can achieve with Hadoop. To ensure resilience of key business processes, many practitioners prefer separating their Hadoop-powered business applications from

developmental or exploratory activities that may either crash the cluster or place it under unpredictable load. In most cases, running two independent clusters is the easiest way to ensure this separation.

- **Competing business priorities**. As multiple business units within an organization come to depend upon Hadoop, they place competing demands upon it and the infrastructure underpinning a cluster. When budgets are devolved and competing demands cannot routinely be prioritized amicably, organizations may choose to fall back upon the simple solution of giving different departments control of their own cluster.

Additional situations in which separate Hadoop clusters might even be considered essential include:

- **Disaster Recovery (DR).** Organizations may wish to maintain two separate but synchronized Hadoop clusters so that the backup facility can complete key workloads in the event of a critical failure in the primary data center. Companies like CloudVelox (formerly CloudVelocity) are increasingly delivering cloud-based DR solutions, but the majority of DR investment continues to be in the provisioning of replicated physical infrastructure.

- **Regulatory requirements.** Industries such as financial services, insurance, health care, and pharmaceuticals are heavily regulated, with national and international requirements governing much of the way in which they conduct business. These regulations can include provisions requiring data to be stored in specific countries or regions, insisting that data related to one business activity be kept separate from data related to another, or even mandating that demonstrably identical copies of data be maintained in different locations. Any or all of these may mean that a company must operate more than one Hadoop cluster.

- **Latency.** Hadoop workloads involve the ingest, processing, and analysis of very large data volumes. Even within a single data center, with high-speed local area networks optimized to cope, this can present challenges. But for organizations operating internationally, the challenges of continually and repeatedly moving significant data volumes over long distances may prove prohibitive. In these situations, operating separate Hadoop clusters in each region of interest and keeping data processing as close as possible to the point at which data is created or stored may make sense.

But what about the data lake?

Organizations have a variety of reasons to consider operating more than one Hadoop cluster. One significant challenge is the very real danger of data duplication and creating silos of data, something that the whole concept of the data lake was meant to resolve.

If Hadoop clusters are managed completely independently of one another, then their underlying HDFS data stores will by default also be separate. In some situations, this is considered a benefit of running separate Hadoop clusters, but in most cases it simply creates an additional problem: finding cost-effective and reliable ways to keep data in sync.

Keeping data in sync

Ensuring that data on different, often distant systems remains in sync is a problem for more than just Hadoop. An entire industry has emerged to offer tools and processes that address the challenges with varying degrees of success. The data volumes typically involved in Hadoop workloads increase the complexity here, but several options exist for replicating data between Hadoop clusters with a reasonable degree of success:

- Low-level tools like rsync offer a way to incrementally copy files from one filesystem to another. Simple checksums can be used to verify that data has moved without being changed, but with the volume of data a typical Hadoop cluster has to contend with, the processing overhead of continually running checksums and reconciling differences between systems may prove prohibitive. In extreme cases, demonstrably reconciled copies may lag hours or even days behind the master data, which is unlikely to be acceptable for most Hadoop use cases.

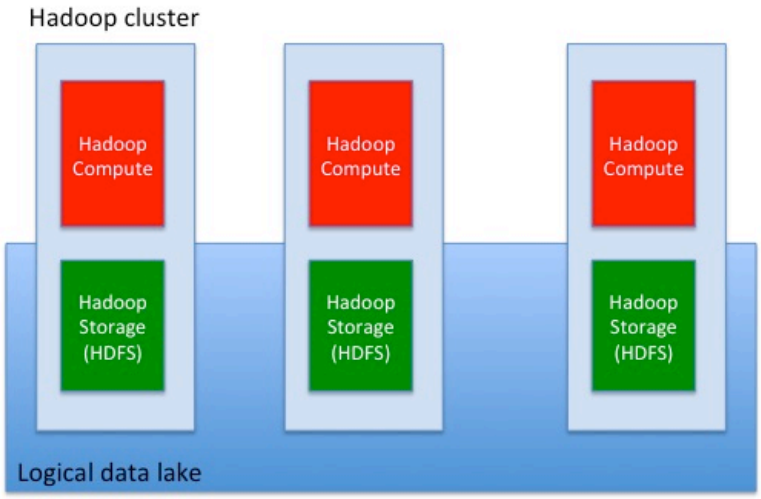
- Within Hadoop itself, the distributed copy (DistCp) tool is intended to support inter- and intra-cluster movement of data. Although mostly reliable, it does suffer from some limitations. It may, for example, fail to recognize that a file has changed (and therefore needs to be replicated) if its name and filesize remain the same as before. Breaks in network connectivity between sites can also introduce errors that DistCp will struggle to recover from. The basic process of copying large files from one node to another may place undue stress on the network and impact cluster performance.

- For data entering Hadoop from outside, existing Hadoop-related projects like Flume and Kafka can be used to ensure that incoming data is immediately written to two or more separate locations. Both tools are well suited to data streaming from remote sensors or to importing log data from

external monitoring processes. Neither is really optimized to support movement of data within or between clusters.

An alternative approach is to pool the HDFS storage from multiple Hadoop clusters, creating a logical data lake of shared storage accessed by compute resources that remain isolated and discrete within their own originating cluster.

A logical data lake



Source: Gigaom Research

Burstable compute

One advantage of this approach is that it creates opportunities to put otherwise idle backup clusters to work. Where a physical DR facility is used, money, power, and network bandwidth are already consumed to ensure that the Hadoop storage within the DR facility is kept in sync with the primary cluster it is meant to replicate. The compute nodes within the DR facility are essentially unused until a problem occurs in the primary cluster and operations fail over to the DR facility.

Within a logical data lake, storage in the DR facility is fully readable *and* writable. It receives updates from the primary cluster and is capable of contributing local changes back to the primary cluster. In these circumstances, customers may choose to put the idle compute nodes to work processing data that can then be made available to other parts of the business. This could happen as either a routine occurrence or, perhaps more plausibly, a cost-effective way to meet occasional spikes in demand for compute resources. Some companies already have customers in highly regulated industries such as financial services that are running proof-of-concept deployments based on this approach. These customers appear satisfied that the approach continues to meet their regulatory requirements around redundancy, backup, and disaster recovery provision.

Separate compute, shared storage

The same approach can be applied to the challenge of workload optimization. As discussed above, YARN is responsible for allocating workloads to available resources within a Hadoop cluster. However, in its current form YARN is not always capable of effectively allocating specific workloads to the most appropriate compute resources. Important or time-critical jobs may well end up running on lower-performance hardware within the cluster. Tasks that benefit from large quantities of memory or faster network performance may be allocated to machines without those characteristics, while less demanding workloads run unnecessarily on the most capable hardware.

With all of the storage pooled in a logical data lake, it becomes feasible to create smaller or more targeted compute clusters effectively. One might be optimized for streaming data, another for in-memory processing, and a third for lower priority tasks. All three share the same storage, and all three can read from or write to the lake to ensure that the organization retains a single comprehensive view across its data.

Next steps

Hadoop workloads are becoming more complex, more nuanced, and more important to the organizations running them. Existing cluster management capabilities such as those in YARN or workflow schedulers like Oozie cope increasingly well with mainstream deployments that fill a single Hadoop cluster with a pool of broadly equivalent hardware resources. However they struggle with some of the use cases described in this report, such as DR, workload optimization, and addressing regulatory requirements. These use cases, and others like them, show every indication of becoming more common than they are today.

Core Hadoop components like YARN, related projects like Oozie, and Apache incubator projects waiting to begin will, in all likelihood, eventually gain the ability to properly address some or all of today's edge-case workloads. Growing demand from potential beneficiaries, particularly in heavily regulated sectors, will no doubt drive further development activity within the community, but it will take time for new projects to gain traction. Opportunities exist, particularly in the short- to medium-term, for commercial providers to step in and deliver solutions tailored to the needs of demanding and deep-pocketed customers who already recognize this requirement.

Key Takeaways

Recent innovations in the core Hadoop project, such as YARN, mean that Hadoop clusters are increasingly capable of cost-effectively and efficiently running a mix of workloads that extend far beyond Hadoop's MapReduce-based origins. This improvement in efficiency and technical capability encourages those operating Hadoop clusters to use them in richer and more complex ways, extracting greater value from their investment and increasing the number of applications able to share a growing pool of readily accessible data.

This pool of shared data — increasingly referred to as a data lake — is becoming a persuasive attribute of the Hadoop-based approach, with customers of the principal Hadoop vendors recognizing the value of escaping their current data silos in order to derive insight from data generated across the organization.

However, despite Hadoop's advances, customers still have some compelling reasons, at least some of the time, to continue managing multiple Hadoop clusters:

- The desire to separate development and production workloads, or to optimize workloads for particular hardware profiles
- Regulatory concerns around colocation of data or workflows
- Latency challenges associated with moving data across global distances

In these and similar situations, the rationale for operating separate clusters may be self-evident, but the resulting fragmentation of the data lake is something that operators of these clusters appear keen to avoid.

Given the current absence of fully developed solutions from within the Hadoop project itself, early adopters are turning to partial solutions based upon low-level file replication techniques, and commercial providers are stepping in to fill the gap with proprietary solutions intended to address the use cases highlighted by their early customers.

About Paul Miller

Paul Miller is an analyst and consultant, based in the East Yorkshire (U.K.) market town of Beverley and working with clients worldwide. He helps clients understand the opportunities (and pitfalls) around cloud computing, big data, and open data, as well as presenting, podcasting, and writing for a number of industry channels. His background includes public policy and standards roles, several years in senior management at a U.K. software company, and a Ph.D. in archaeology.

About Gigaom Research

Gigaom Research gives you insider access to expert industry insights on emerging markets. Focused on delivering highly relevant and timely research to the people who need it most, our analysis, reports, and original research come from the most respected voices in the industry. Whether you're beginning to learn about a new market or are an industry insider, Gigaom Research addresses the need for relevant, illuminating insights into the industry's most dynamic markets.

Visit us at: research.gigaom.com.