Compliments of IBM Platform Computing

Hadoop® FOR DUMMIES®

Special Edition

Making Everything Easier!™

FREE eTips at dummies.com®

Robert D. Schneider

Hadoop® FOR DUMMIES® SPECIAL EDITION

Hadoop®
FOR DUMMIES®
SPECIAL EDITION

by Robert D. Schneider

WILEY

John Wiley & Sons, Inc.

Hadoop For Dummies®, Special Edition

Published by
John Wiley & Sons Canada, Ltd.
6045 Freemont Blvd.
Mississauga, ON L5R 4J3
www.wiley.com

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

For details on how to create a custom book for your company or organization, or for more information on John Wiley & Sons Canada custom publishing programs, please call 416-646-7992 or email publishingbyobjectives@wiley.com**.**

Wiley publishes in a variety of print and electronic formats and by print-on-demand. For more information about Wiley products, visit www.wiley.com.

ISBN: 978-1-118-25051-8

Printed in the United States

1 2 3 4 5 DPI 17 16 15 14 13

WILEY

About the Author

Robert D. Schneider is a Silicon Valley–based technology consultant and author. He has provided database optimization, distributed computing, and other technical expertise to a wide variety of enterprises in the financial, technology, and government sectors.

He has written six books and numerous articles on database technology and other complex topics such as cloud computing, Big Data, data analytics, and Service Oriented Architecture (SOA). He is a frequent organizer and presenter at technology industry events, worldwide. Robert blogs at `http://rdschneider.com`.

Special thanks to Rohit Valia, Jie Wu, and Steven Sit of IBM for all of their help in reviewing this book.

Publisher's Acknowledgments

We're proud of this book; please send us your comments at http://dummies.custhelp.com.

Some of the people who helped bring this book to market include the following:

Acquisitions and Editorial

Associate Acquisitions Editor: Anam Ahmed

Production Editor: Pauline Ricablanca

Copy Editor: Heather Ball

Editorial Assistant: Kathy Deady

Composition Services

Project Coordinator: Kristie Rees

Layout and Graphics: Jennifer Creasey

Proofreader: Jessica Kramer

John Wiley & Sons Canada, Ltd.

Deborah Barton, Vice President and Director of Operations

Jennifer Smith, Publisher, Professional and Trade Division

Alison Maclean, Managing Editor, Professional and Trade Division

Publishing and Editorial for Consumer Dummies

Kathleen Nebenhaus, Vice President and Executive Publisher

David Palmer, Associate Publisher

Kristin Ferguson-Wagstaffe, Product Development Director

Publishing for Technology Dummies

Richard Swadley, Vice President and Executive Group Publisher

Andy Cummings, Vice President and Publisher

Composition Services

Debbie Stailey, Director of Composition Services

Contents at a Glance

Table of Contents

Introduction

Welcome to *Hadoop For Dummies!* Today, organizations in every industry are being showered with imposing quantities of new information. Along with traditional sources, many more data channels and categories now exist. Collectively, these vastly larger information volumes and new assets are known as *Big Data*. Enterprises are using technologies such as MapReduce and Hadoop to extract value from Big Data. The results of these efforts are truly mission-critical in size and scope. Properly deploying these vital solutions requires careful planning and evaluation when selecting a supporting infrastructure.

In this book, we provide you with a solid understanding of key Big Data concepts and trends, as well as related architectures, such as MapReduce and Hadoop. We also present some suggestions about how to implement high-performance Hadoop.

Foolish Assumptions

Although taking anything for granted is usually unwise, we do have some expectations of the readers of this book.

First, we surmise that you have some familiarity with the colossal amounts of information (also called Big Data) now available to the modern enterprise. You also understand what's generating this information as well as how it's being used. Examples of today's data sources consist of traditional enterprise software applications along with many new channels and categories such as weblogs, sensors, mobile devices, images, audio, and so on. Relational databases, data warehouses, and sophisticated business intelligence tools are among the most common consumers of all this information.

Next, we infer that you are either in technical or line-of-business management (with a title such as a chief information officer, director of IT, operations executive, and so on), or that you have hands-on experience with Big Data through an architect, database administrator, or business analyst role.

Finally, regardless of your specific title, we assume that you're interested in making the most of the mountains of information that are now available to your organization. We also figure that you want to do all of this in the most scalable, high-performance, and secure manner possible.

How This Book Is Organized

The five chapters in this book equip you with everything you need to understand the benefits and drawbacks of various solutions for Big Data, along with how to optimally deploy MapReduce and Hadoop technologies in your enterprise:

- **Chapter 1, Introducing Big Data:** Provides some background about the explosive growth of unstructured data and related categories, along with the challenges that led to the introduction of MapReduce and Hadoop.
- **Chapter 2, MapReduce to the Rescue:** Explains how MapReduce offers a fresh approach to gleaning value from the vast quantities of data that today's enterprises are capturing and maintaining.
- **Chapter 3, Hadoop: MapReduce for Everyone:** Illustrates why generic, out-of-the-box MapReduce isn't suitable for most organizations. Highlights how the Hadoop stack provides a comprehensive, end-to-end, ready for prime time MapReduce implementation.
- **Chapter 4, Enterprise-grade Hadoop Deployment:** Describes the special needs of production-grade Hadoop MapReduce implementation.
- **Chapter 5, Ten Tips for Getting the Most from Your Hadoop Implementation:** Lists a collection of best practices that will maximize the value of your Hadoop experience.

Icons Used in This Book

Every For Dummies book has small illustrations, called icons, sprinkled throughout the margins. We use these icons in this book.

TIP

This icon guides you to right-on-target information to help you get the most out of your Hadoop software.

REMEMBER

This icon highlights concepts worth remembering as you immerse yourself in MapReduce and Hadoop.

TECHNICAL STUFF

If you'd like to explore the next level of detail, be on the lookout for this icon.

ON THE WEB

Seek out this icon if you'd like to learn even more about Big Data, MapReduce, and Hadoop.

Chapter 1

Introducing Big Data

In This Chapter

- Beginning with Big Data
- Meeting MapReduce
- Saying hello to Hadoop
- Making connections between Big Data, MapReduce, and Hadoop

There's no way around it: learning about Big Data means getting comfortable with all sorts of new terms and concepts. This can be a bit confusing, so this chapter aims to clear away some of the fog.

What Is Big Data?

The first thing to recognize is that Big Data does not have one single definition. In fact, it's a term that describes at least three separate, but interrelated, trends:

- **Capturing and managing lots of information:** Numerous independent market and research studies have found that data volumes are doubling every year. On top of all this extra new information, a significant percentage of organizations are also storing three or more years of historic data.
- **Working with many new types of data:** Studies also indicate that 80 percent of data is *unstructured* (such as images, audio, tweets, text messages, and so on). And until recently, the majority of enterprises have been unable to take full advantage of all this unstructured information.

- **Exploiting these masses of information and new data types with new styles of applications:** Many of the tools and technologies that were designed to work with relatively large information volumes haven't changed much in the past 15 years. They simply can't keep up with Big Data, so new classes of analytic applications are reaching the market, all based on a next generation Big Data platform. These new solutions have the potential to transform the way you run your business.

Driving the growth of Big Data

Just as no single definition of Big Data exists, no specific cause exists for what's behind its rapid rate of adoption. Instead, several distinct trends have contributed to Big Data's momentum.

New data sources

Today, we have more generators of information than ever before. These data creators include devices such as mobile phones, tablet computers, sensors, medical equipment, and other platforms that gather vast quantities of information.

Traditional enterprise applications are changing, too: e-commerce, finance, and increasingly powerful scientific solutions (such as pharmaceutical, meteorological, and simulation, to name a few) are all contributing to the overall growth of Big Data.

Larger information quantities

As you might surmise from its name, Big Data also means that dramatically larger data volumes are now being captured, managed, and analyzed.

To demonstrate just how much bigger Big Data can be, consider this: Over a history that spans more than 30 years, SQL database servers have traditionally held *gigabytes* of information — and reaching that milestone took a long time. In the past 15 years, data warehouses and enterprise analytics expanded these volumes to *terabytes*. And in the last five years, the distributed file systems that store Big Data now routinely house *petabytes* of information. As we describe later, all of this new data has placed IT organizations under great stress.

New data categories

How does your enterprise's data suddenly balloon from gigabytes to hundreds of terabytes and then on to petabytes? One way is that you start working with entirely new classes of information. While much of this new information is relational in nature, much is not. In the past, most relational databases held records of complete, finalized transactions. In the world of Big Data, *sub-transactional data* plays a big part, too, and here are a few examples:

- Click trails through a website
- Shopping cart manipulation
- Tweets
- Text messages

Relational databases and associated analytic tools were designed to interact with *structured information* — the kind that fits in rows and columns. But much of the information that makes up today's Big Data is *unstructured* or *semi-structured*, such as these examples:

- Photos
- Video
- Audio
- XML documents

XML documents are particularly interesting: they form the backbone of many of today's enterprise applications, yet have proven very demanding for earlier generations of analytic tools to cope with. This is partially because of XML's habitually massive size, and partially because of its semi-structured nature.

Commoditized hardware and software

The final piece of the Big Data puzzle is the low-cost hardware and software environments that have recently become so popular. These innovations have transformed technology, particularly in the last five years. As we see later, capturing and exploiting Big Data would be much more difficult and costly without the contributions of these cost-effective advances.

Differentiating between Big Data and traditional enterprise relational data

Thinking of Big Data as "just lots more enterprise data" is tempting, but it's a serious mistake. First, Big Data is notably larger — often by several orders of magnitude. Secondly, Big Data is commonly generated outside of traditional enterprise applications. And finally, Big Data is often composed of unstructured or semi-structured information types that continually arrive in enormous amounts.

To get maximum value from Big Data, it needs to be associated with traditional enterprise data, automatically or via purpose-built applications, reports, queries, and other approaches. For example, a retailer might want to link its Web site visitor behavior logs (a classic Big Data application) with purchase information (commonly found in relational databases). In another case, a mobile phone provider might want to offer a wider range of smartphones to customers (inventory maintained in a relational database) based on text and image message volume trends (unstructured Big Data).

Knowing what you can do with Big Data

Big Data has the potential to revolutionize the way you do business. It can provide new insights into everything about your enterprise, including the following:

- The way your customers locate and interact with you
- The way you deliver products and services to the marketplace
- The position of organization vs. your competitors
- Strategies you can implement to increase profitability
- And many more

What's even more interesting is that these insights can be delivered in real-time, but only if your infrastructure is designed properly.

Big Data is also changing the analytics landscape. In the past, structured data analysis was the prime player. These tools and techniques work well with traditional relational database-hosted information. In fact, over time an entire industry has grown around structured analysis. Some of the most notable players include SAS, IBM (Cognos), Oracle (Hyperion), and SAP (Business Objects).

Driven by Big Data, unstructured data analysis is quickly becoming equally important. This fresh exploration works beautifully with information from diverse sources such as wikis, blogs, Facebook, Twitter, and web traffic logs.

To help bring order to these diverse sources, a whole new set of tools and technologies is gaining traction. These include MapReduce, Hadoop, Pig, Hive, Hadoop Distributed File System (HDFS), and NoSQL databases.

Checking out challenges of Big Data

As is the case with any exciting new movement, Big Data comes with its own unique set of obstacles that you must find a way to overcome, such as these barriers:

- **Information growth:** Over 80 percent of the data in the enterprise consists of unstructured data, which tends to be growing at a much faster pace than traditional relational information. These massive volumes threaten to swamp all but the most well-prepared IT organizations.
- **Processing power:** The customary approach of using a single, expensive, powerful computer to crunch information just doesn't scale for Big Data. As we soon see, the way to go is divide-and-conquer using commoditized hardware and software via scale-out.
- **Physical storage:** Capturing and managing all this information can consume enormous resources, outstripping all budgetary expectations.
- **Data issues:** Lack of data mobility, proprietary formats, and interoperability obstacles can all make working with Big Data complicated.

- **Costs:** Extract, transform, and load (ETL) processes for Big Data can be expensive and time consuming, particularly in the absence of specialized, well-designed software.

These complications have proven to be too much for many Big Data implementations. By delaying insights and making detecting and managing risk harder, these problems cause damage in the form of increased expenses and diminished revenue.

Consequently, computational and storage solutions have been evolving to successfully work with Big Data. First, entirely new programming frameworks can enable distributed computing on large data sets, with MapReduce being one of the most prominent examples. In turn, these frameworks have been turned into full-featured product platforms such as Hadoop.

There are also new data storage techniques that have arisen to bolster these new architectures, including very large file systems running on commodity hardware. One example of a new data storage technology is HDFS. This file system is meant to support enormous amounts of structured as well as unstructured data.

While the challenge of storing large and often unstructured data sets has been addressed, providing enterprise-grade services to work with all this data is still an issue. This is particularly prevalent with open-source implementations.

What Is MapReduce?

As we describe in this chapter, old techniques for working with information simply don’t scale to Big Data: they’re too costly, time-consuming, and complicated. Thus, a new way of interacting with all this data became necessary, which is where MapReduce comes in.

In a nutshell, MapReduce is built on the proven concept of divide and conquer: it’s much faster to break a massive task into smaller chunks and process them in parallel.

Dividing and conquering

While this concept may appear new, in fact there's a long history of this style of computing, going all the way back to LISP in the 1960s.

Faced with its own set of unique challenges, in 2004 Google decided to bring the power of parallel, distributed computing to help digest the enormous amounts of data produced during daily operations. The result was a group of technologies and architectural design philosophies that came to be known as MapReduce.

Check out `http://research.google.com/archive/mapreduce.html` to see the MapReduce design documents.

In MapReduce, task-based programming logic is placed as close to the data as possible. This technique works very nicely with both structured and unstructured data. It's no surprise that Google chose to follow a divide-and-conquer approach, given its organizational philosophy of using lots of commoditized computers for data processing and storage instead of focusing on fewer, more powerful (and expensive!) servers. Along with the MapReduce architecture, Google also authored the Google File System. This innovative technology is a powerful, distributed file system meant to hold enormous amounts of data. Google optimized this file system to meet its voracious information processing needs. However, as we describe later, this was just the starting point.

Google's MapReduce served as the foundation for subsequent technologies such as Hadoop, while the Google File System was the basis for the Hadoop Distributed File System.

Witnessing the rapid rise of MapReduce

If only Google was deploying MapReduce, our story would end here. But as we point out earlier in this chapter, the explosive growth of Big Data has placed IT organizations in every industry under great stress. The old procedures for handling all this information no longer scale, and organizations needed a new approach. *Parallel processing* has proven to be an excellent way of coping with massive amounts of input data. Commodity hardware and software makes it cost-effective to employ hundreds or thousands of servers — working in parallel — to answer a question.

MapReduce is just the beginning: it provides a well-validated technology architecture that helps solve the challenges of Big Data, rather than a commercial product per se. Instead, MapReduce laid the groundwork for the next subject that we discuss: Hadoop.

What Is Hadoop?

MapReduce is a great start, but it requires you to expend a significant amount of developer and technology resources to make it work in your organization. This isn't feasible for most enterprises unless the name on the office building says "Google." This relative complexity led to the advent of Hadoop.

Hadoop is a well-adopted, standards-based, open-source software framework built on the foundation of Google's MapReduce and Google File System papers. It's meant to leverage the power of massive parallel processing to take advantage of Big Data, generally by using lots of inexpensive commodity servers.

Hadoop is designed to abstract away much of the complexity of distributed processing. This lets developers focus on the task at hand, instead of getting lost in the technical details of deploying such a functionally rich environment.

The not-for-profit Apache Software Foundation has taken over maintenance of Hadoop, with Yahoo! making significant contributions. Hadoop has gained tremendous adoption in a wide variety of organizations, including the following:

- Social media (e.g., Facebook, Twitter)
- Life sciences
- Financial services
- Retail
- Government

We describe the exact makeup of Hadoop later in the book.

For now, remember that your Hadoop implementation must have a number of qualities if you're going to be able to rely on it for critical enterprise functionality:

- **Application compatibility:** Given that the Hadoop implementation of MapReduce is meant to support the entire enterprise, you must choose your Hadoop infrastructure to foster maximum interoperability. You'll want to search for solutions with these features:
    - Open architecture with no vendor lock-in
    - Compatibility with open standards
    - Capability of working with multiple programming languages
- **Heterogeneous architecture:** Your Hadoop environment must be capable of consuming information from many different data sources — both traditional as well as newer. Since Hadoop also stores data, your goal should be to select a platform that provides two things:
    - Flexibility when choosing a distributed file system
    - Data independence from the MapReduce programming model
- **Support for service level agreements (SLA):** Since Hadoop will likely be powering critical enterprise decision-making, be sure that your selected solution can deliver in these areas:
    - High predictability
    - High availability
    - High performance
    - High utilization
    - High scalability
    - Worldwide support
- **Latency requirements:** Your Hadoop technology infrastructure should be adept at executing different types of jobs without too much overhead. You should be able to prioritize processes based on these features:

  - Need for real time
  - Low latency — less than one millisecond
  - Batch

- **Economic validation:** Even though Hadoop will deliver many benefits to your enterprise, your chosen technology should feature attractive total cost of ownership (TCO) and return on investment (ROI) profiles.

Seeing How Big Data, MapReduce, and Hadoop Relate

The earlier parts of this chapter describe each of these important concepts — Big Data, MapReduce, and Hadoop. So here's a quick summary of how they relate:

- **Big Data:** Today most enterprises are facing lots of new data, which arrives in many different forms. Big Data has the potential to provide insights that can transform every business. And Big Data has spawned a whole new industry of supporting architectures such as MapReduce.
- **MapReduce:** A new programming framework — created and successfully deployed by Google — that uses the divide-and-conquer method (and lots of commodity servers) to break down complex Big Data problems into small units of work, and then process them in parallel. These problems can now be solved faster than ever before, but deploying MapReduce alone is far too complex for most enterprises, which led to Hadoop.
- **Hadoop:** A complete technology stack that implements the concepts of MapReduce to exploit Big Data. Hadoop has also spawned a robust marketplace served by open-source and value-add commercial vendors. As we describe later in this book, you absolutely must research the marketplace to make sure that your chosen solution will meet your enterprise's needs.

Chapter 2

MapReduce to the Rescue

In This Chapter

- Knowing why MapReduce is essential
- Understanding how MapReduce works
- Looking at the industries that use MapReduce
- Considering real-world applications

MapReduce — originally created by Google — has proven to be a highly innovative technique for taking advantage of the huge volumes of information that organizations now routinely process.

In this chapter, we begin by explaining the realities that drove Google to create MapReduce. Then we move on to describe how MapReduce operates, the sectors that can benefit from its capabilities, and real scenarios of MapReduce in action.

Why Is MapReduce Necessary?

REMEMBER

In the past, working with large information sets would have entailed acquiring a handful of extremely powerful servers. Each of these machines would have very fast processors and lots of memory. Next, you would need to stage massive amounts of high-end, often proprietary storage. You'd also be writing big checks to license expensive operating systems, relational database management systems (RDBMS), business intelligence, and other software. To put all of this together, you would hire highly skilled consultants. All in all, this effort takes lots of time and money.

Because this whole process was so complex, expensive, and brittle, enterprises frequently restricted interaction with the resulting solutions. These constraints were tolerable when the amount of data was measured in gigabytes, and the internal user community was small. Of course, if the ignored users complained loudly enough, the organization might find a way to throw more time and money at the problem and grant additional access to the coveted information resources.

This scenario no longer scales in today's world. Nowadays, data is measured in terabytes to petabytes and data growth rates exceed 25 percent each year. In turn, a significant percentage of this data is unstructured. Meanwhile, increasing numbers of users are clamoring for access to all this information. Fortunately, technology industry trends have applied fresh techniques to work with all this information:

- Commodity hardware
- Distributed file systems
- Open source operating systems, databases, and other infrastructure
- Significantly cheaper storage
- Service-oriented architecture

However, while these technology developments addressed part of the challenges of working with Big Data, no well-regarded, proven software architecture was in place. So Google — faced with making sense of the largest collection of data in the world — took on this challenge. The result was MapReduce: a software framework that breaks big problems into small, manageable tasks and then distributes them to multiple servers. Actually, "multiple servers" is an understatement; hundreds of computers may contain the data needing to be processed. These servers are called *nodes*, and they work together in parallel to arrive at a result.

MapReduce is a huge hit. Google makes very heavy use of MapReduce internally, and the Apache Software Foundation turned to MapReduce to form the foundation of its Hadoop implementation.

Check out the Apache Hadoop home page at `http://hadoop.apache.org` to see what the fuss is all about.

How Does MapReduce Work?

In this section, we examine the workflow that drives MapReduce processing. To begin, we explain how much data you need to have before benefitting from MapReduce's unique capabilities. Then it's on to MapReduce's architecture, followed by an example of MapReduce in action.

How much data is necessary to use MapReduce?

If you're tasked with trying to gain insight into a relatively small amount of information (such as hundreds of megabytes to a handful of gigabytes), MapReduce probably isn't the right approach for you. For these types of situations, many time-tested tools and technologies are suitable. On the other hand, if your job is to coax insight from a very large disk-based information set — often measured in terabytes to petabytes — then MapReduce's divide-and-conquer tactics will likely meet your needs.

MapReduce can work with raw data that's stored in disk files, in relational databases, or both. The data may be structured or unstructured, and is commonly made up of text, binary, or multi-line records. Weblog records, e-commerce click trails, and complex documents are just three examples of the kind of data that MapReduce routinely consumes.

The most common MapReduce usage pattern employs a distributed file system known as *Hadoop Distributed File System* (HDFS). Data is stored on local disk and processing is done locally on the computer with the data.

MapReduce architecture

At its core, MapReduce is composed of two major processing steps: Map and Reduce. Put them together, and you've got MapReduce. We look at how each of these steps works.

Map

In contrast with traditional relational database–oriented information — which organizes data into fairly rigid rows and columns that are stored in tables — MapReduce uses *key/value pairs*.

As you might guess from their name, each instance of a key/value pair is made up of two data components. First, the *key* identifies what kind of information we're looking at. When compared with a relational database, a key usually equates to a column.

Easily understood instances of keys include

- First name
- Transaction amount
- Search term

Next, the *value* portion of the key/value pair is an actual instance of data associated with a key. Using the brief list of key examples from above, relevant values might include

- Danielle
- 19.96
- Snare drums

If you put the keys and values together, you end up with key/value pairs:

- First name/Danielle
- Transaction amount/19.96
- Search term/Snare drums

In the Map phase of MapReduce, records from the data source are fed into the map() function as key/value pairs. The map() function then produces one or more intermediate values along with an output key from the input.

Reduce

After the Map phase is over, all the intermediate values for a given output key are combined together into a list. The reduce() function then combines the intermediate values into one or more final values for the same key.

This is a much simpler approach for large-scale computations, and is meant to abstract away much of the complexity of parallel processing. Yet despite its simplicity, MapReduce lets you crunch massive amounts of information far more quickly than ever before.

Configuring MapReduce

When setting up a MapReduce environment, you need to consider this series of important assumptions:

- **Components will fail at a high rate:** This is just the way things are with inexpensive, commodity hardware.
- **Data will be contained in a relatively small number of big files:** Each file will be 100 MB to several GB.
- **Data files are write-once:** However, you are free to append these files.
- **Lots of streaming reads:** You can expect to have many threads accessing these files at any given time.
- **Higher sustained throughput across large amounts of data:** Typical Hadoop MapReduce implementations work best when consistently and predictably processing colossal amounts of information across the entire environment, as opposed to achieving irregular sub-second response on random instances. However, in some scenarios mixed workloads will require a *low latency solution* that delivers a near real-time environment. IBM Big Data solutions provide a choice of batch, low-latency or real-time solutions to meet the most demanding workloads.

MapReduce in action

To help you visualize the concepts behind MapReduce, we consider a realistic example. In this scenario, you're in charge of the e-commerce website for a very large retailer. You stock over 200,000 individual products, and your website receives hundreds of thousands of visitors each day. In aggregate, your customers place nearly 50,000 orders each day.

Over time, you've amassed an enormous collection of search terms that your visitors have typed in on your website. All of this raw data measures several hundred terabytes. The marketing department wants to know what customers are interested in, so you need to start deriving value from this mountain of information.

Starting with a modest example, the first project is simply to come up with a sorted list of search terms. Here's how you will apply MapReduce to produce this list:

1. The data should ideally be broken into numerous 1 GB +/- files.
2. Each file will be distributed to a different node.
3. On each node, the Map step will produce a list, consisting of each word in the file along with how many times it appears. For example, one node might come up with these intermediate results from its own set of data:

   …

   Skate: 4992120

   Ski: 303021

   Skis: 291101

   …
4. The Reduce step will then consolidate all of the results from the Map step, producing a list of all search terms and the total number of times they appeared across all of the files. For example, the combined counts for these search terms might look like this:

   …

   Skate: 1872695210

   Ski: 902785455

   Skis: 3486501184

   …

Who Uses MapReduce?

The short answer to this question is — anyone! You only need three things to benefit from MapReduce:

- **Lots of data:** This will probably exceed several terabytes.
- **Multiple servers at your disposal:** These can be on premise, in the cloud, or both.
- **MapReduce-based software – such as Hadoop**: Later on, we make suggestions you can use to pick the right Hadoop technology to meet your needs.

MapReduce techniques have been successfully deployed in a wide variety of vertical markets, such as these:

- ✓ Financial services
- ✓ Telco
- ✓ Retail
- ✓ Government
- ✓ Defense
- ✓ Homeland security
- ✓ Health and life services
- ✓ Utilities
- ✓ Social networks/Internet
- ✓ Internet service providers

There's no limit to what you can do with MapReduce. Here are just a few instances of these MapReduce applications.

- ✓ Risk modeling
- ✓ Recommendation engines
- ✓ Point of sale transaction analysis
- ✓ Threat analysis
- ✓ Search quality
- ✓ ETL logic for data warehouses
- ✓ Customer churn analysis
- ✓ Ad targeting
- ✓ Network traffic analysis
- ✓ Trade surveillance
- ✓ Data sandboxes

Real-World MapReduce Examples

TIP

To help bring the potential of MapReduce to life, here are a few actual scenarios taken from three very different industries. In each segment, we describe several situations in which MapReduce has been employed.

Financial services

Given the amount of money that's traded each day, this industry has some of the most rigorous processing needs. Market pressures as well as regulatory mandates drive these requirements. MapReduce is being used in each of the illustrations we're about to describe.

Fraud detection

Traditional algorithms based on sampling alone aren't sufficient to capture rare events and prevent fraud. Instead, these firms need better prediction and fraud prevention with greater capabilities to analyze large volumes of data. The MapReduce framework is effective for pattern recognition applications, which are often used in fraud detection. MapReduce allows the financial institution to recognize unusual trading activity and to flag it for human review.

Asset management

Financial organizations must contend with millions of instruments, which in turn lead to billions of individual records. MapReduce can perform a broad range of tasks suitable for maintaining these assets, including these:

- Broker volume analysis
- Sector level metrics
- Trade activities analysis
- Trade cost analysis

Data source and data store consolidation

Financial firms now commonly offer an enormous range of services, such as mortgages, credit and debit cards, banking transactions, calls centers, trading desks, and so on. Increased regulatory requirements mandate that historical data be kept longer. Data mining and governance practices have also been strengthened. MapReduce drives the applications necessary to store, manage, analyze, and safeguard all of this information.

Retail

Faced with razor-thin margins, retailers rely on MapReduce to enhance profitability while staying ahead of the competition.

Web log analytics

Even a mid-sized retailer's website will generate gargantuan amounts of weblog data. MapReduce is ideal for analyzing consumer purchasing data to identify buying behaviors and look for patterns to help design targeted marketing programs.

Improving customer experience and improving relevance of offers

Retailers can use MapReduce results to help link structured enterprise data with unstructured search and site traffic information. For example, merchandising analysts can combine structured product pricing information with unstructured shopping cart manipulation details to promote higher-margin products more effectively.

Supply chain optimization

Confronted by relentless profitability pressures, retailers are always looking to increase their margins. MapReduce helps analyze data generated during the supply chain process. The results of these computations can then be applied towards streamlining how, where, and when inventory is acquired.

Life sciences

Genomic research and analysis — including complex tasks such as DNA assembly and mapping, variant calling, and other annotation processes — requires powerful storage and computational resources for high volume pipelines.

Whole-genome analysis of unstructured genomic data requires linking observations about genomes back to established results from databases and research literature. The results help researchers connect the dots between disease and clinical treatment pathway options.

MapReduce is proving to be a promising new programming technique to effectively and efficiently analyze the complex unstructured genomic data.

Auto manufacturing

Whether they build stately land yachts or the newest, most fuel-efficient vehicles, every auto manufacturer must cope with a highly complex business environment. Here are a few MapReduce solutions that can help.

Vehicle model and option validation

MapReduce algorithms can be applied to calculating how to fabricate a base vehicle using a vast library of configuration data. This lets the manufacturer evaluate multiple models, and then define and validate which options can be installed on a particular vehicle. MapReduce can also be used to make sure that these selections will be compatible with each other, as well as to ensure that no laws or internal regulations are broken in the vehicle's design.

Vehicle mass analysis

Creating a new car model is a highly complex undertaking. MapReduce can be used to create many "what if" algorithms. These formulas can then be used to scrutinize design concepts under multiple load conditions that can then be tested through digital prototyping and optimization tools.

Emission reporting

Vehicle manufacturers are required by law to periodically execute complex calculations on topics such as vehicle pollution generated on a given trip (carbon monoxide, NOx, hydrocarbons, CO_2) as well as monthly CO_2 emissions and carbon footprints. MapReduce can quickly crunch the enormous amounts of data necessary to produce these results.

Customer satisfaction

Since vehicle manufacturers sell millions of cars, their customer satisfaction surveys yield formidable quantities of raw information. MapReduce is the perfect approach to glean intelligence from all of this data. It can also be used to close the loop between warranty claims and future vehicle designs. Much of this data is structured and unstructured, which is another point in favor of MapReduce.

Chapter 3

Hadoop: MapReduce for Everyone

In This Chapter

- Going beyond MapReduce
- Meeting Hadoop
- Looking at Hadoop's architecture
- Evaluating Hadoop's ecosystem
- Comparing open-source vs. commercial implementations

When Google created MapReduce, they designed a distributed, parallel processing architecture uniquely qualified to exploit Big Data's potential. In this chapter, we point out that while MapReduce is a great start, the next logical step in its evolution is Hadoop: an open-source software solution meant for widespread adoption. We explore Hadoop's architecture and ecosystem, and list some potential challenges for Hadoop in the enterprise.

Why MapReduce Alone Isn't Enough

Unless your employer's name happens to be Google, you probably don't have enough internal software developers and system administrators to implement and maintain the entire infrastructure MapReduce needs. For that matter, you probably made the decision long ago not to design and implement your own proprietary operating system, RDBMS, application server, and so on.

So a need existed for a complete, standardized, end-to-end solution suitable for enterprises that wanted to apply MapReduce techniques to get value from reams of Big Data — which is where Hadoop comes in.

Introducing Hadoop

The first question you're probably asking is, "What's a Hadoop?" Believe it or not, the original Hadoop is a toy elephant, specifically, the toy elephant belonging to Doug Cutting's son. Doug — who created the Hadoop implementation of MapReduce — gave this name to the new initiative.

After it was created, Hadoop was turned over to the Apache Software Foundation. Hadoop is now maintained as an open-source, top-level project with a global community of contributors. It's written in Java, and its original deployments include some of the most well-known (and most technically advanced) organizations, such as:

- Yahoo!
- Facebook
- LinkedIn
- Netflix
- And many others

Hadoop cluster components

A typical Hadoop environment is generally made up of a master node along with worker nodes. In turn, each of these nodes consists of several specialized software components, which we describe next.

Master node

The majority of Hadoop deployments consist of several master node instances. Having more than one master node helps eliminate the risk of a single point of failure. As we describe later, the Hadoop solutions supplied by best-of-breed commercial vendors go further and use additional techniques to help augment its overall reliability.

Here are major elements present in the *master node*:

- **JobTracker:** This process is assigned to interact with client applications. It is also responsible for distributing MapReduce tasks to particular nodes within a cluster.
- **TaskTracker:** This is a process in the cluster that is capable of receiving tasks (including Map, Reduce, and Shuffle) from a JobTracker.
- **NameNode:** These processes are charged with storing a directory tree of all files in the Hadoop Distributed File System (HDFS). They also keep track of where the file data is kept within the cluster. Client applications contact NameNodes when they need to locate a file, or add, copy, or delete a file.

DataNodes

The *DataNode* stores data in the HDFS, and is responsible for replicating data across clusters. DataNodes interact with client applications when the NameNode has supplied the DataNode's address.

Worker nodes

Unlike the master node, whose numbers you can usually count on one hand, a representative Hadoop deployment consists of dozens or even hundreds of *worker nodes*, which provide enough processing power to analyze a few hundred terabytes all the way up to one petabyte. Each worker node includes a DataNode as well as a TaskTracker.

Hadoop Architecture

Envision a Hadoop environment as consisting of three basic layers. These represent a logical hierarchy with a full separation of concerns. Together, these levels deliver a full MapReduce implementation. We inspect each layer in more detail.

Application layer/end user access layer

This stratum provides a programming framework for applying distributed computing to large data sets. It serves as the point of contact for applications to interact with Hadoop. These applications may be internally written solutions, or third-party tools such as business intelligence, packaged enterprise software, and so on.

To build one of these applications, you normally employ a popular programming interface such as Java, Pig (a specialized, higher-level MapReduce language), or Hive (a specialized, SQL-based MapReduce language).

Learn more about Apache Pig at `http://pig.apache.org` and visit `http://hive.apache.org` for more about Apache Hive.

MapReduce workload management layer

Commonly known as JobTracker, this Hadoop component supplies an open-source runtime job execution engine. This engine coordinates all aspects of your Hadoop environment, such as scheduling and launching jobs, balancing the workload among different resources, and dealing with failures and other issues. Scheduling itself is frequently performed by software developed from the Apache Oozie project.

As we tell you later, this layer is the most critical for guaranteeing enterprise-grade Hadoop performance and reliability.

Distributed parallel file systems/data layer

This layer is responsible for the actual storage of information. For the sake of efficiency, it commonly uses a specialized distributed file system. In most cases, this file system is HDFS (Hadoop Distributed File System).

Along with HDFS, this layer may also consist of commercial and other third-party implementations. These include IBM's GPFS, the MapR filesystem from MapR Technologies, Kosmix's CloudStore, and Amazon's Simple Storage Service (S3).

The inventors of the HDFS made a series of important design decisions:

- **Files are stored as blocks:** These are much larger than most file systems, with a default of 128 MB.
- **Reliability is achieved through replication:** Each block is replicated across two or more DataNodes; the default value is three.
- **A single master NameNode coordinates access and metadata:** This simplifies and centralizes management.
- **No data caching:** It's not worth it given the large data sets and sequential scans.
- **There's a familiar interface with a customizable API:** This lets you simplify the problem and focus on distributed applications, rather than performing low-level data manipulation.

While HDFS is a reasonable choice for a file system, in a moment we describe a number of its limitations for enterprise-grade Hadoop implementations.

Hadoop's Ecosystem

REMEMBER

To serve this growing market, many vendors now offer a complete Hadoop distribution that includes all three layers we discuss under "Hadoop Architecture." These distributions have commonly been modified and extended to solve specific challenges, such as availability, performance, and application-specific use cases. Many best-of-breed suppliers also exist, each with a particular niche in the Hadoop stack.

Layers and players

In this section, we look at how these vendors have addressed five distinct Hadoop specialties. To make this list as comprehensive as possible, we cover niche players as well as major suppliers. Since the distributed MapReduce runtime is so important to enterprise-grade Hadoop, we also devote a special section to some tips you can use to select a provider. Finally, we point out some of the advantages and disadvantages of commercial versus open-source suppliers.

Distributed data storage

These technologies are used to maintain large datasets on commodity storage clusters. Given that this information forms the foundation of your MapReduce efforts, be sure that your Hadoop implementation is capable of working with all types of data storage.

Major providers include the following:

- Hadoop HDFS
- IBM GPFS
- Appistry CloudIQ Storage
- MapR Technologies

Distributed MapReduce runtime

This software is assigned an extremely important responsibility: scheduling and distributing jobs that consume information kept in distributed data storage. The following are some major suppliers:

- Open-source Hadoop JobTracker
- IBM Platform Symphony MapReduce
- Oracle Grid Engine
- Gridgain

Supporting tools and applications

A broad range of technologies lets programmers and nonprogrammers alike derive value from Big Data, such as these well-known examples:

- Programming tools:
    - Apache Pig
    - Apache Hive
- Workflow scheduling:
    - Apache Oozie
- Data store:
    - Apache HBase

- Analytic and related tools:
  - IBM BigSheets
  - Datameer
  - Digital Reasoning

Distributions

These provide a single, integrated offering of all components, pre-tested and certified to work together. Here are the most illustrious of the many providers:

- Apache Hadoop
- IBM InfoSphere BigInsights
- Cloudera
- Hortonworks
- MapR Technologies

Business intelligence and other tools

These are popular technologies that have been in use for years working with traditional relational data. They've now been extended to work with data that's accessible via Hadoop. Here are three industry sub-segments, along with some of the best-known vendors in each one:

- Analytics
  - IBM Cognos
  - IBM SPSS
  - MicroStrategy
  - Quest
  - SAS
  - Jaspersoft
  - Pentaho
- Extract, transform, load (ETL)
  - IBM InfoSphere DataStage
  - Informatica
  - Pervasive
  - Talend

- Data warehouse
    - IBM Netezza
    - Oracle
    - Greenplum
    - Teradata

Evaluation criteria for distributed MapReduce runtimes

In the next chapter, we tell you more about how choosing the right MapReduce runtime is a critical responsibility. For now, here are some high-level questions you should ask.

MapReduce programming APIs

- Is the solution compatible with Apache's Hadoop?
- Can it be extended to additional languages?

Job scheduling and workload management

- Can the solution schedule jobs that are made up of Map and Reduce tasks?
- Does it support policies such as:
    - FIFO
    - Capacity
    - Fair share scheduling
    - Pre-emptive scheduling
    - Threshold scheduling
- Can it balance cluster utilization with workload prioritization?

Scalable distributed execution management

- Can the solution manage task execution on remote compute nodes?
- What is the performance throughput and overhead for running jobs and tasks on large clusters?

Data affinity and awareness

- Can the solution place tasks on compute nodes based on data location?
- Does it have failover and other reliability safeguards for storage?
- Does it have communication with the storage layer (i.e., NameNode)?
- Does it supply an HDFS compatible interface?

Resource management

- Does the solution have awareness/management capabilities for resources that are available in the cluster?
- Does the solution offer a flexible mechanism for lending and borrowing resources at runtime?
- Is the resource management function capable of supporting heterogeneous type of workloads running on the common set of resources?

Job/task failover and availability

- Can the solution perform automatic failover to rerun jobs or tasks that have failed?
- Will submitted tasks be automatically recovered if a failure of the Job Tracker node/process occurs?

Operational management and reporting

- Does it offer monitoring of the environment, including jobs and resources?
- Does the solution provide statistics on historical performance for reporting and chargeback?
- Can jobs be paused and restarted?

Debugging and troubleshooting

- Does the solution offer a centralized collection of application, job, and task logs?

Application lifecycle management deployment and distribution

- Does the solution provide mechanisms for deploying
    - User programs into the cluster?
    - Rolling application upgrades without requiring cluster shutdown for maintenance?
- Does it have support for multiple application versions on the same cluster?

Support for multiple application types

- Can MapReduce and other types of application coexist on the same cluster?
- Is there support for C, C++, common scripting languages, and so on?

Support for multiple lines of business

- Can the solution provide guaranteed Service Level Agreements (SLA) to multiple lines of business that are sharing the resources?
- Can line-of-business managers administer their own separate queues while running on a shared grid infrastructure?

Open-source vs. commercial Hadoop implementations

As we described earlier, Yahoo! created Hadoop as an open-source project. Google's MapReduce and the Google File System (GFS) inspired this project. When Hadoop was complete, Yahoo! turned it over to the Apache Software Foundation.

Open-source challenges

Many of the issues with open source revolve around the amount of work you must do to deploy and maintain these implementations. A significant chance of performance and reliability problems is also a consideration. Here are some of the most troubling open-source concerns:

- **Hadoop design flux:** Hadoop's source code is continually evolving. This isn't surprising, given the global contributor community along with the uniform architecture of original Hadoop platform.

- **Deployment:** Hadoop open-source sites struggle with a lack of options, such as being required to deploy it as a single, monolithic application with restricted file system support. Customers must implement and support the entire Hadoop stack themselves. It's Java-centric and primarily works with the Hadoop file system. These restrictions introduce the possibility of wasted or underused resources.
- **Daily operations:** Open-source Hadoop is meant for IT departments with substantial internal resources and expertise. Furthermore, its workload management layer introduces a single point of failure and potential for business disruptions.

  When a failure does occur, the cycle for job re-start is long, often requiring manual job resurrection and troubleshooting. These limitations often manifest in substandard performance along with delayed results. So daily operations are hampered by minimal management, scalability, and availability capabilities. The end result is diminished SLA capabilities, and difficulties meeting business requirements from multiple lines of business.

Commercial challenges

As is the case with many other popular open-source projects, numerous commercial open-source vendors have arrived to service the Hadoop market. Customers must evaluate these choices carefully, not only for current usage patterns but also for potential expansion.

In-data warehouse

The primary problem with this approach is that these types of Hadoop implementations typically only work with their own data warehouses. Usually very little support is available for unstructured data files such as those found in HDFS, IBM GPFS, and so on. Finally, standard support may be spotty, so be on the lookout for solutions that comply with these industry guidelines.

Full Hadoop stack

Vendors such as IBM, Cloudera, Hortonworks, and MapR Technologies all supply a complete Hadoop stack. These platforms offer integrated, well-tested, enterprise-class solutions. However, even these offerings may present some problems:

- By default these solutions are based on the Hadoop JobTracker and don't offer some of the capabilities required by the production environment.
- Diminished support for managing multiple/mixed workloads.
- Resource utilization is tuned for larger jobs. However, some platforms — such as IBM InfoSphere BigInsights — have the ability to optimize shorter workloads.

Customers can benefit from the power of open-source Hadoop for enterprise deployments by leveraging the IBM InfoSphere BigInsights and IBM Platform Symphony solution that we describe in the next chapter.

Chapter 4

Enterprise-grade Hadoop Deployment

In This Chapter

- Listing essential Hadoop characteristics
- Picking the right commercial Hadoop implementation

Earlier chapters talk about what makes up Big Data, and how MapReduce and Hadoop are helping all sorts of organizations get the most out of all this new information.

As the number of Hadoop installations continues to grow, these deployments are expected to provide mission-critical capabilities. However, the only way for these essential projects to succeed in production environments is if the underlying Hadoop software is capable of meeting stringent performance, security, reliability, and scalability requirements. This is no different than what's necessary for other core technologies, such as operating systems, database servers, or web servers.

In this chapter, we provide you with a collection of guidelines to ensure that your Hadoop technology meets your needs.

High-Performance Traits for Hadoop

Before we get started on some recommendations for your Hadoop implementation, remember that a typical Hadoop installation consists of three primary layers (which we discuss in Chapter 3):

- **Application layer:** This provides user and developer-accessible APIs for working with Big Data.
- **Workload management layer:** This layer is charged with dividing your Map and Reduce tasks and distributing them onto available resources. It coordinates interaction with your data and is also the most likely place where the mission-critical capabilities of your Hadoop technology will come into play.
- **Data layer:** This is responsible for storing and accessing all of the information with which your Hadoop environment will interact.

REMEMBER

Each of these layers presents its own unique performance, scalability, and security challenges that you must address if you want to derive full value from your Hadoop environment.

If you need to rely on your Hadoop architecture to underpin critical business operations, make sure that your selected implementation technology offers the following:

- **High scalability:** It should enable deployment and operation of the extraction and analysis programs across large numbers of nodes with minimal overhead, even as the number of nodes keeps increasing.
- **Low latency:** Many Hadoop applications require a low-latency response to enable the quickest possible results for shorter jobs. Unfortunately, open-source Hadoop imposes high overhead on job deployment, making short-run jobs pay a heavy performance price. With this in mind, be sure to select a low-latency Hadoop architecture if your workload mix will contain short-running jobs.
- **Predictability:** Your workload scheduler should provide a policy-based, sophisticated scheduling mechanism. Ideally, it will offer different algorithms to satisfy varying needs. This type of scheduling engine increases certainty and permits the IT team to support guaranteed SLAs for multiple lines of business.
- **High availability:** Your Hadoop environment should supply workload management that offers quality of service (QoS) and automatic failover capabilities.
- **Easy management:** This is especially important given that many Hadoop implementations involve large-scale deployments.

- **Multi-tenancy:** A single high-performance, production-ready Hadoop implementation must be capable of servicing multiple MapReduce projects through one consolidated management interface. Each of these initiatives is likely to have a different business use case as well as diverse performance, scale, and security requirements.

The Hadoop workload management layer makes many of these traits possible. Not coincidentally, this is the layer where most open-source implementations fail to keep pace with customer demands.

Choosing the Right Hadoop Technology

IT organizations in financial services, life sciences, government, oil and gas, and many other industries are faced with new challenges stemming from the growth and increased retention of data. In addition, not only are these groups required to manage their infrastructure on smaller budgets, they're also increasingly being asked to deliver incremental services, especially for getting value from data.

The most successful organizations find a way to transform raw data into intelligence. This rule is true whether the enterprise is tasked with defending the nation or simply responding to competitive pressures in the marketplace. Sophisticated analytics performed on Big Data is proving to be a valuable tool to garner insight that existing tools can't match.

REMEMBER

IBM provides a comprehensive portfolio of customizable offerings that help organizations get the most out of Big Data. IBM InfoSphere BigInsights delivers immediate benefits for customers wishing to get started with a Hadoop implementation. In addition to a full Hadoop distribution, it includes an array of common tools that will accelerate the development of Big Data applications.

ON THE WEB

Check out `http://ibm.com/software/data/infosphere/biginsights` for more about IBM InfoSphere BigInsights.

IBM Platform Symphony Advanced Edition delivers additional capabilities to supply low-latency and multi-tenancy capabilities to your Hadoop environments. It also allows you to build a shared service infrastructure for non-Hadoop applications to execute on the same cluster, and provides sophisticated scheduling and predictable SLAs for your enterprise deployment.

ON THE WEB

Check out `http://ibm.com/platformcomputing/products/symphony` for more details about IBM Platform Symphony.

Chapter 5

Ten Tips for Getting the Most from Your Hadoop Implementation

In This Chapter

- ▶ Setting the foundation for a successful Hadoop environment
- ▶ Managing and monitoring your Hadoop implementation
- ▶ Creating applications that are Hadoop-ready

Throughout this book, we tell you all about Big Data, MapReduce, and Hadoop. This chapter puts all of this information to work by presenting you with a series of field-tested guidelines and best practices.

We begin with some tips for the planning stages of your Hadoop/MapReduce initiative. Next up are some recommendations about how to build successful Hadoop applications. Finally, we offer some ideas about how to roll out your Hadoop implementation for maximum effectiveness.

Involve All Affected Constituents

Successfully deploying Hadoop MapReduce is a team effort. Since Hadoop will affect a wide range of people in your organization, be sure to inform and get feedback from everyone involved. Here are just a few of these people:

- ✓ Data scientist
- ✓ Database architect/administrator

- Data warehouse architect/administrator
- System administrator/IT manager
- Storage administrator/IT manager
- Network administrator
- Business owner

Determine How You Want To Cleanse Your Data

Traditionally, IT organizations have expended significant resources on one-time cleansing of their data before turning it over to business intelligence and other analytic applications.

In the MapReduce world, however, consider leaving your data in place and cleansing it as part of each task. This preserves information that may be of use later, or for different jobs.

Determine Your SLAs

Service Level Agreements (SLAs) are what IT organizations use to help justify budgets, staffing, and so on. Meet your SLAs, and you're a hero. Miss your SLAs, and the results aren't pretty.

Since SLAs are so vital, and Hadoop applications can vary so widely, you'll need to configure your Hadoop infrastructure to properly serve each constituency. This is likely to entail multiple SLAs, with each one based on priority and other factors. Ask these questions for each SLA:

- How many jobs will be run?
- What kinds of applications will make up these jobs?
- What will their priority be? For example, will these jobs be run overnight, or in real-time?
- How will data growth, security requirements, and expected availability impact your SLAs?

Come Up with Realistic Workload Plans

TIP

Every Hadoop implementation is distinctive: what satisfies a bank is very different than what a retailer needs. For that matter, a single Hadoop environment will likely need to sustain multiple workloads. The only way that you can realistically support this variety is to document and then carefully plan for each type of workload.

For each workload, you need to know many details:

- ✓ User counts
- ✓ Data volumes
- ✓ Data types
- ✓ Processing windows (e.g., a few seconds, minutes or hours)
- ✓ Anticipated network traffic
- ✓ Applications that will consume Hadoop results

This information will affect your infrastructure, software architecture, and so on.

Plan for Hardware Failure

Commodity hardware is a double-edged sword. On one hand, these inexpensive servers make it possible (and affordable) to create powerful, sophisticated MapReduce environments. On the other hand, one sad fact about commodity hardware is that it’s subject to relatively frequent failure.

When you consider just how many nodes make up a Hadoop environment, and then factor in the unreliability of this inexpensive hardware, you can see why these failures happen all the time. Consequently, you should treat hardware failure as an everyday occurrence, and then architect for it.

Focus on High Availability for HDFS

The HDFS NameNode has the potential to be a single point of failure, although this will be addressed in an upcoming version of Hadoop. This vulnerability can disrupt your well-planned MapReduce processing. You can take several approaches to increase the overall availability of HDFS. Some are based on hardware, using overlapping clusters, for example. Others are based on software, with secondary NameNodes and BackupNodes as the approach.

Choose an Open Architecture That Is Agnostic to Data Type

Since you're likely to have data in many types of locations (HDFS, flat files, RDBMS), select technology that is designed to work with heterogeneous data. This offers superior flexibility by making adding new data sources easier. It also lowers your application maintenance burden.

Host the JobTracker on a Dedicated Node

Charged with distributing workloads throughout the Hadoop environment, the JobTracker is an essential component of your MapReduce implementation. Poor throughput here will affect the overall application performance.

For the best possible performance from your JobTracker, be sure to segregate it onto its own machine, and also make sure that this dedicated server is stocked with plenty of memory.

The Hadoop Job Tracker has a single process to manage both workload distribution and resource management. This architecture has inherent limitations. Future implementations are working to resolve this.

IBM Platform Symphony delivers a production-ready solution that implements this next generation architecture for MapReduce applications driving greater infrastructure sharing and utilization.

Configure the Proper Network Topology

Since MapReduce relies on divide-and-conquer principles, extensive amounts of data will normally be moved between the Map and Reduce stages. Any bottlenecks — such as a sluggish network or excessive distance between nodes — will greatly diminished performance.

Putting proper networking in place is a great way to facilitate this important step. When designing your network, think about how mappers coincide with reducers. Try to avoid making unnecessary hops through routers. Stay on on the same rack where possible.

Employ Data Affinity Wherever Possible

TIP

In MapReduce, the goal is to bring the computational power to the data, instead of shipping massive amounts of raw data across the network. This is particularly important given the amount of data typically processed in a Hadoop MapReduce environment. Employing efficient data affinity techniques can yield big savings on network traffic, which translates to significantly faster performance.

Notes

Notes

Notes

Join the Big Data movement with *Hadoop For Dummies!*

Discover effective data solutions for your organization

In the age of "Big Data", it's essential for any organization to know how to analyze and manage their ever-increasing stores of information. Packed with everything you need to know about Hadoop analytics, this handy guide provides you with a solid understanding of the critical Big Data concepts and trends, and suggests ways for you to revolutionize your business operations through the implementation of cost-effective, high-performance Hadoop technology.

Discover how to:

Develop your business strategy through data analysis

Overcome the challenges of Big Data

Maximize the value of your data with MapReduce and Hadoop

Select the best technology to fit your enterprise's needs

THE DUMMIES WAY®

Explanations in plain English

"Get in, get out" information

Icons and other navigational aids

Top ten lists

A dash of humor and fun

Get smart!

@ **www.dummies.com**

- ✓ *Find listings of all our books*
- ✓ *Choose from many different subject categories*
- ✓ *Sign up for eTips at etips.dummies.com*

ISBN: 978-1-118-25051-8
Not for resale

For Dummies®
A Branded Imprint of
WILEY