Big Data
- Structures

Dr. Qing “Matt” Zhang
ITU

Outline

- Big Data Strategies
  - Shared nothing parallel RDBMS
  - NoSQL (K/V stores)
  - MapReduce
- Big Data and Visualization
- Big Data Storage

Big Data Strategies

Shared Nothing

- **Shared nothing** “except Network” parallel RDBMS. Network needs to be fast at minimum for throughput but preferably for latency as well.
- Partitioning/Sharding:
  - Data distribution among shards
  - ideally little/no inter-shard/inter-node
- Columnar:
  - Useful for aggregation
  - Useful for compression

NoSQL

- **NoSQL (Key/value Stores)**
  - K/V or Document (complex K/V store): MongoDB, CouchDB, etc.
  - Column oriented (tabular K/V store): BigTable, Hbase, Cassandra, Accumulo, etc.
- **K/V Store Characteristics**: relatively low latency, not ACID, and no joins

MapReduce Platform

- MapReduce Platform: general purpose implicit parallel programming paradigm
- Hadoop addresses the CRAP partition
- Is composed of HDFS and map reduce itself
- Not limited to Java streams interface (Python, Ruby, etc.)

MapReduce Platform Example

```
% $HADOOP_HOME/bin/hadoop jar \
$HADOOP_HOME/hadoop-streaming.jar \
-input myInputDirs \
-output /bin/cat \
reducer /bin/wc
```

Big Data and Visualization

Introduction

- Collecting information is no longer a problem, but extracting value from information collections has become progressively more difficult.
- Visualization
  - links the human eye and computer,
  - helps to identify patterns, and
  - extracts insights from large amounts of information

Introduction

- Visualization technology shows considerable promise from increasing the value of large-scales collections of information
- Visualization has been used to
  - communicate ideas
  - monitor trends implicit in data, and
  - explore large volumes of data from hypothesis generation.

Visualization Classification

- Visualization can be classified as
  - scientific visualization,
  - software visualization, and
  - information visualization

Scientific Visualization

- Scientific visualization helps understanding physical phenomena in data
- Mathematical model plays an essential role
- Isosurfaces, volume rendering, and glyphs are commonly used techniques

Scientific Visualization

- Isosurfaces depict the distribution of certain attributes
- Volume rendering allows views to see the entire volume of 3-D data in a single image
- Glyphs provides a way to display multiple attributes through combinations of various visual cues

Software Visualization

- Software visualization helps people understand and use computer software effectively
  - Program visualization helps programmers manage complex software
    - Visualizing the source code, data structure, and the changes made to the software
  - Algorithm **animation** is used to motivate and support the learning of computational algorithms

Information Visualization

- Information visualization helps users identify patterns, correlations, or clusters
  - Structured information
    - Graphical representation to reveal patterns. e.g. Spotfire, SAS/GRAPH, SPSS
    - Integration with various data mining techniques
  - Unstructured Information
    - Need to identify variables and construct visualizable structures. e.g. SemioMap, and Knowledgist

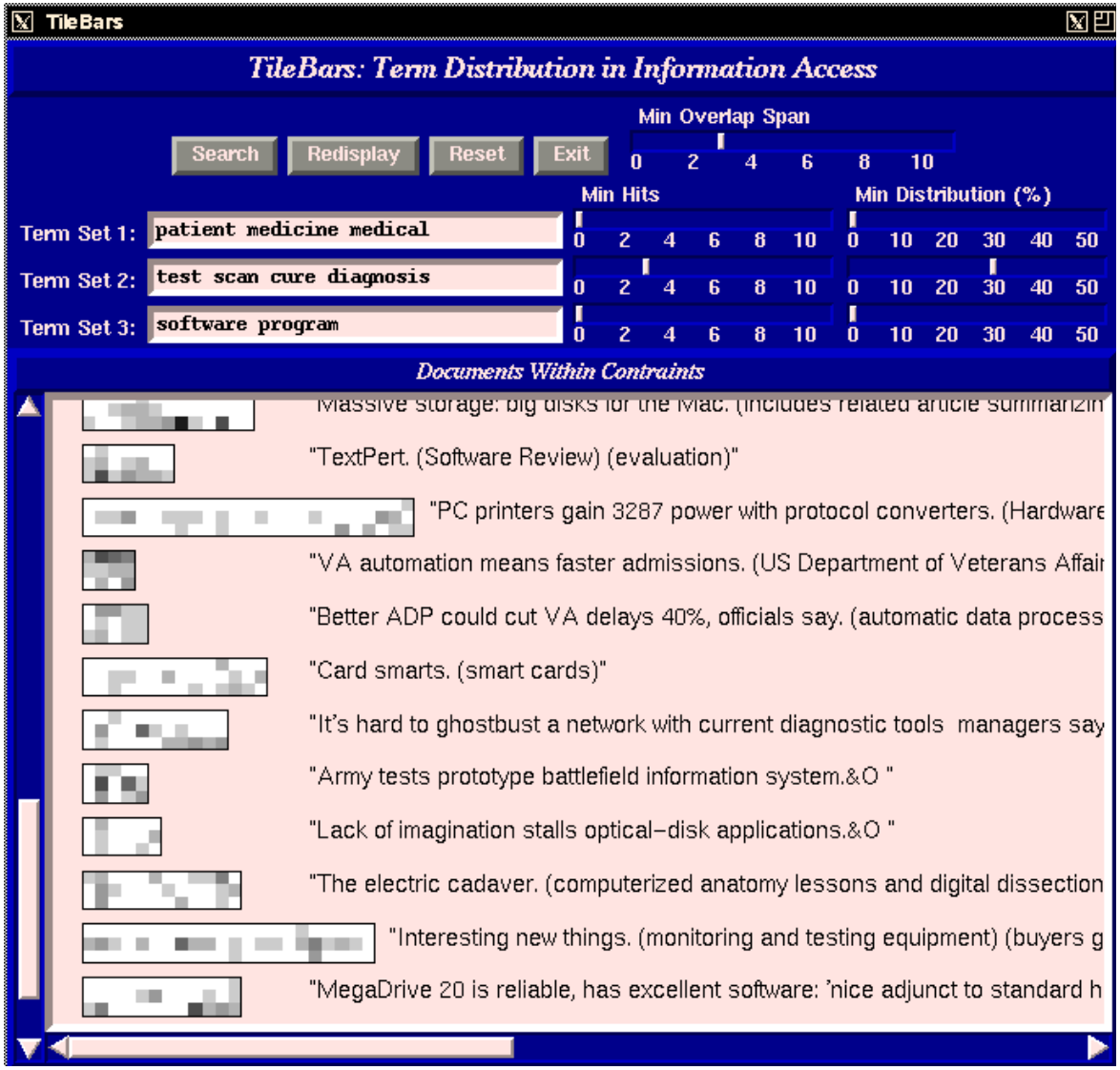
Framework for Information Visualization

- Three research dimensions support the development of an information visualization system
  - Information representation
  - User interface interaction
  - Information analysis

Information Representation

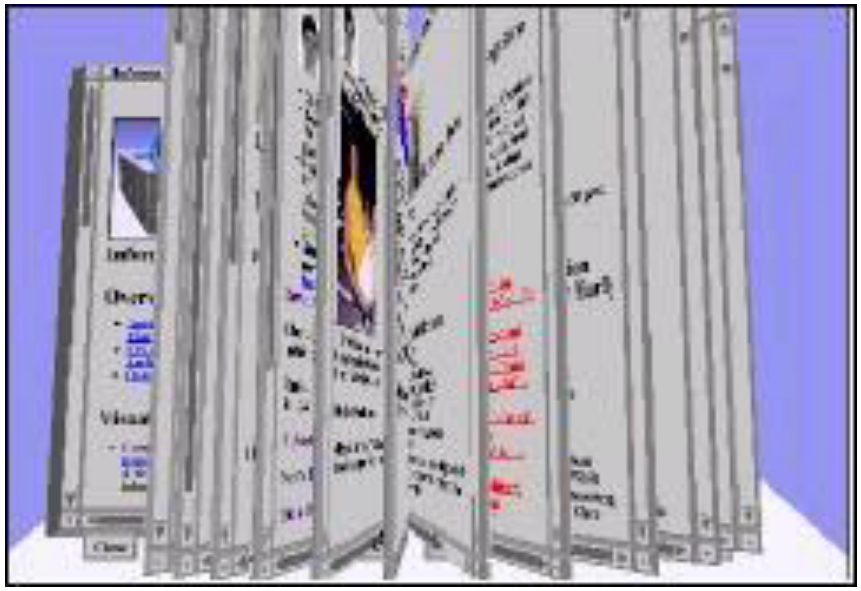
- Seven types of representation methods:
  - 1-D
  - 2-D
  - 3-D
  - Multidimensional
  - Tree
  - Network
  - Temporal
    - Location and animation are common

1-D Example


TileBars
TileBars: Term Distribution in Information Access
Min Overlap Span
Search
Redisplay
Reset
Exit
0 2 4 6 8 10
Min Hits
Min Distribution (%)
Term Set 1: patient medicine medical
Term Set 2: test scan cure diagnosis
Term Set 3: software program
0 2 4 6 8 10
0 10 20 30 40 50
Documents Within Contraints
"TextPert. (Software Review) (evaluation)"
"PC printers gain 3287 power with protocol converters. (Hardware
"VA automation means faster admissions. (US Department of Veterans Affair
"Better ADP could cut VA delays 40%, officials say. (automatic data process
"Card smarts. (smart cards)"
"It's hard to ghostbust a network with current diagnostic tools managers say
"Army tests prototype battlefield information system.&O "
"Lack of imagination stalls optical–disk applications.&O "
"The electric cadaver. (computerized anatomy lessons and digital dissection
"Interesting new things. (monitoring and testing equipment) (buyers g
"MegaDrive 20 is reliable, has excellent software: 'nice adjunct to standard h

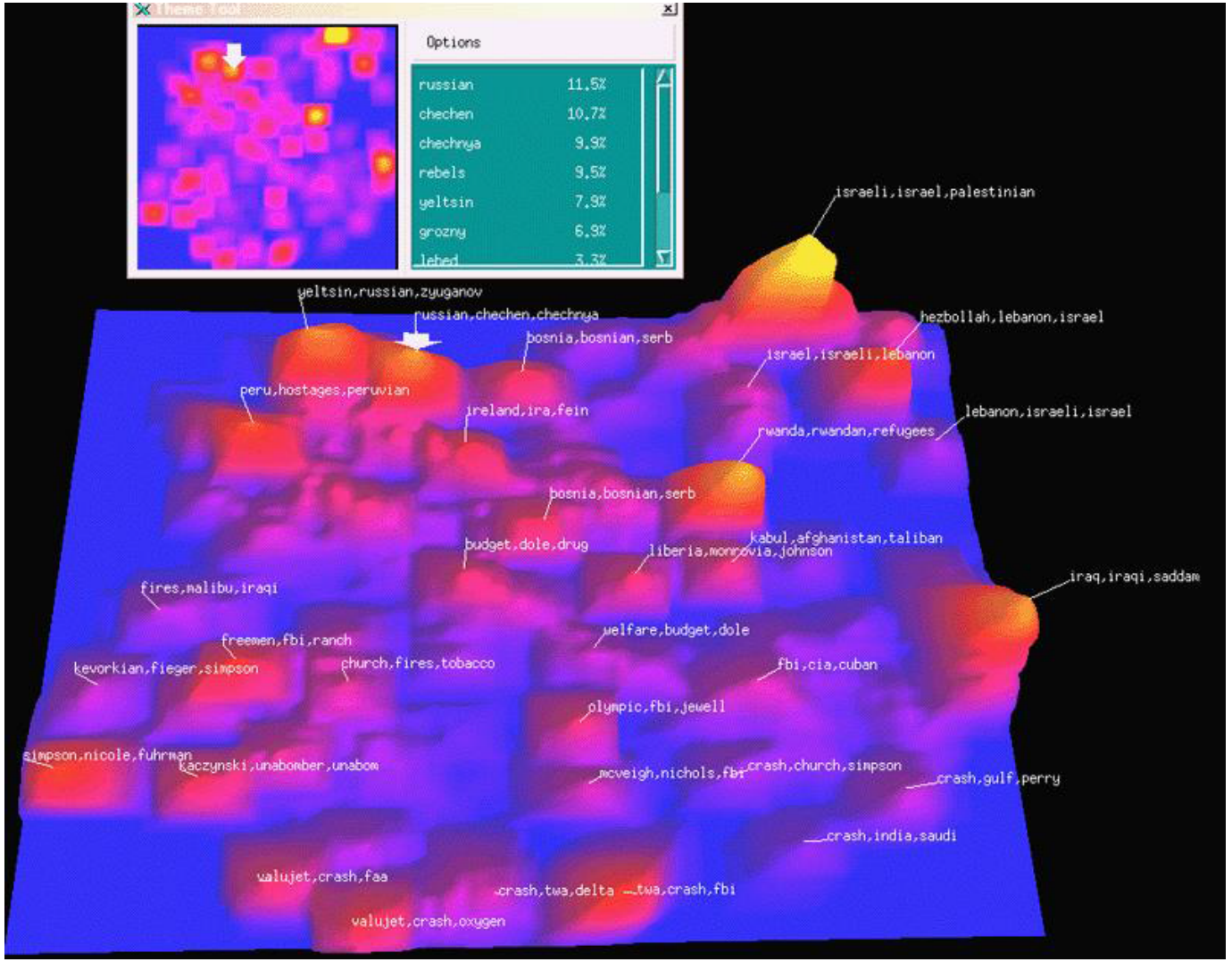
2-D and 3-D

- 2-D Represent information as two-dimensional visual objects
- 3-D To represent information as 3-D visual objects



Multidimensional

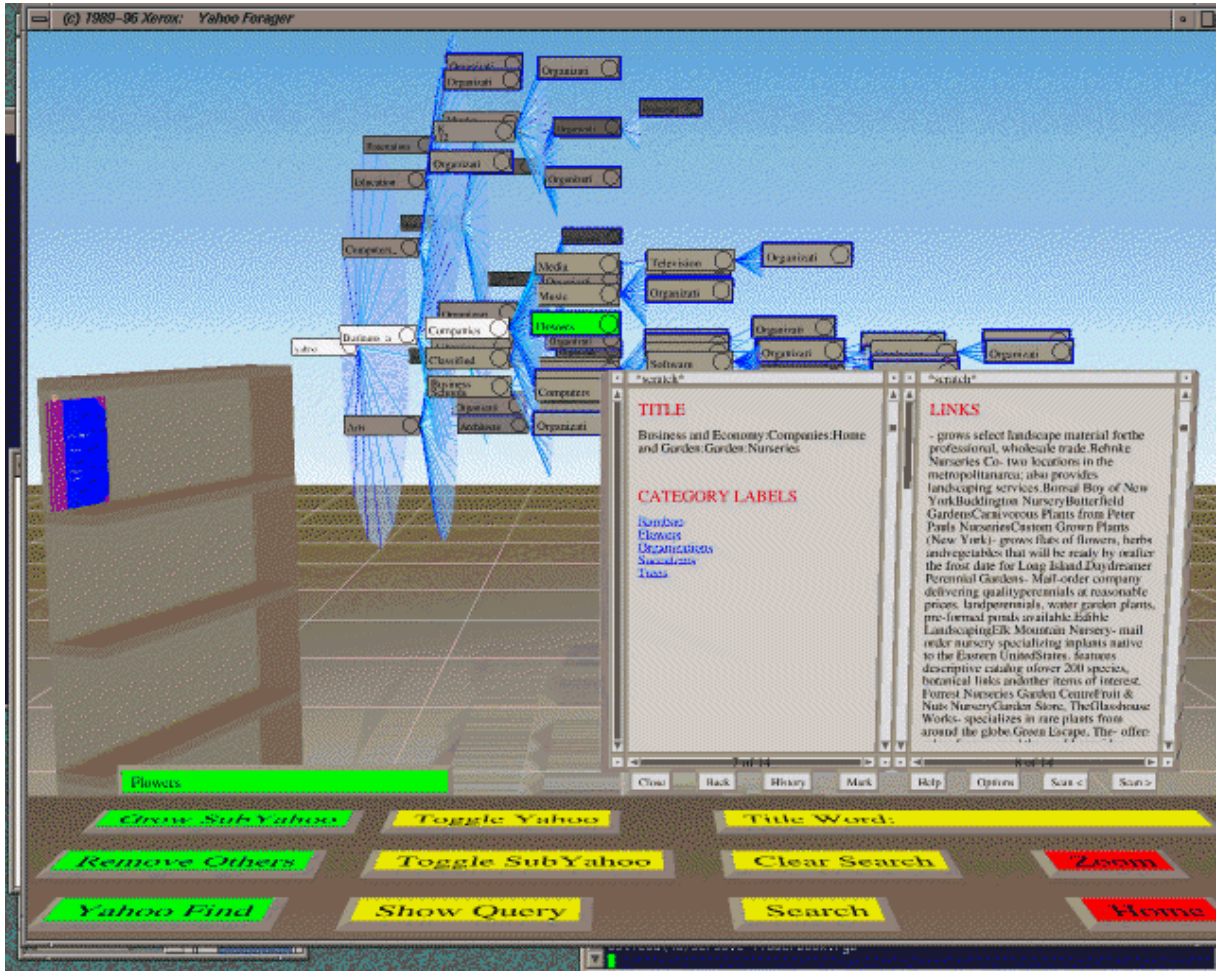
To represent information as multi-dimensional objects and project them into a 3-D or 2-D space



Tree

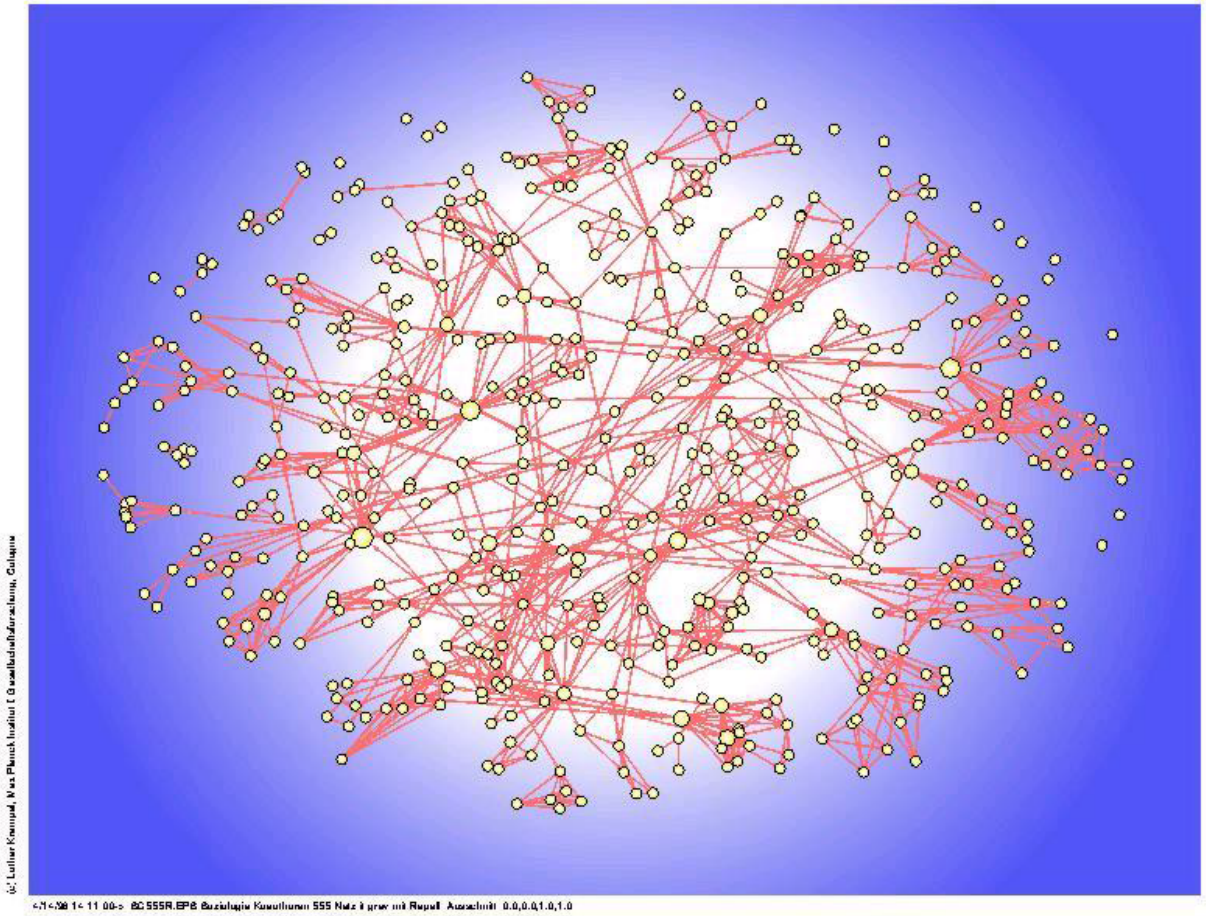
To represent hierarchical relationship

- Challenge: nodes grows exponentially



Network

To represent complex relationships that a simple tree is insufficient

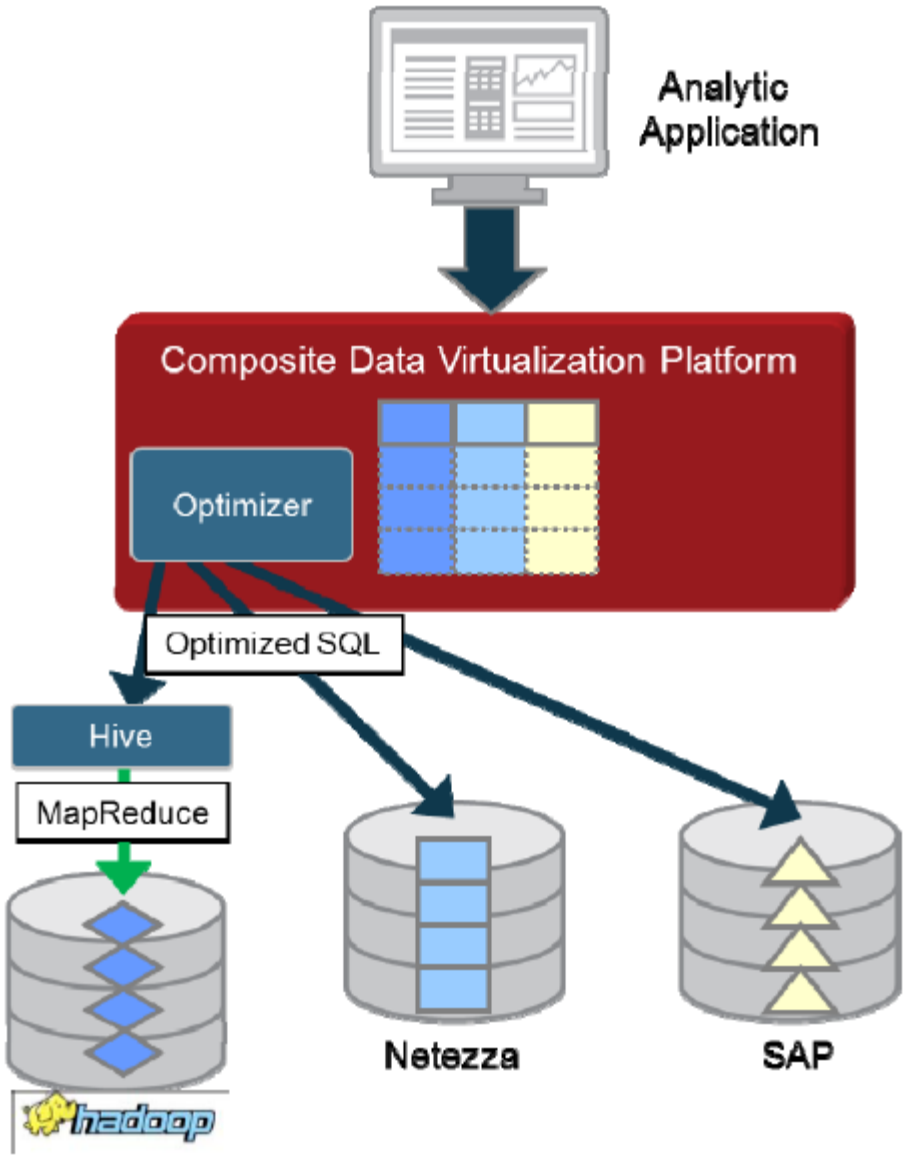


Integration Challenges

- Analytics Push the Limits of Traditional Data Management
- Data Virtualization Solves Big Data Integration Problem
  - The answer to too many silos is not to integrate into a data warehouse because of
    - scalability limitations,
    - replication processing, and
    - storage cost would be too high

Integration Challenges

- The answer is data federation or virtualization



Big Data Storage

Different Roles of Data Consumers

- Software developers need to access data programmatically to build applications
- Some users are less technical and will thus need to access data such as company metrics using a dashboard
- Executives are after high-level understanding of the major trends
- Many employees should not be privy to sensitive human resources information

Implementation Choices

- When faced with the challenge of sharing many gigabytes and even terabytes of data, a variety of potential implementation choices appear
- These choices are influenced by factors such as cost, audience, and expertise.
- Different types of users have different types of data consumer needs

Choosing How to Store the Data

- The first challenge is determining how to physically store as well as enable the ability to share your files in a scalable and economical way
- Although it is simple to just post a bunch of files on any Web server, the cost of storage and bandwidth must scale with the amount of data and the number of users

Choosing the Right Data Format

- The second data-sharing challenge is to decide the format that you should provide for your users.
- This decision depends on your intended audience.
  - Should your files be easy for computer programmers to use, or
  - easy for the average person to upload into a spreadsheet?

Choosing the Right Data Format

- What about space:
  - Should you use the most compact format possible, or
  - optimize for human readability?
- In some cases, you should strive to provide a variety of formats for the various use cases you hope to support

How to Present the Data

- The third consideration to address involves how you allow users to access your data.
- A municipality that aims to share information with its local citizens should provide online data dashboards with well-designed visualizations so that the technically challenged can participate.

How to Present the Data

- However, data journalists and researchers need more than just dashboards
  - they need lots of big, raw, machine-readable data for detailed analysis
- The municipality should also invest in ways to provide programmatic access via Web-based APIs to encourage the development of software applications that consume the data on demand

Storage: Infrastructure as a Service

- Providers like Rackspace, Amazon, Google, and many others have begun to offer services that can be thought of as utility computing, or computing as a service
- The pricing and business model of this type of computing is starting to mimic the models used for public utilities such as power and water companies

Storage: Infrastructure as a Service

- In the infrastructure-as-a-service (IaaS) model, many physical machines are connected together in a network cluster, acting like a single, large computer.
- Data belonging to a single user might be physically stored in small chunks on many different physical machines

Storage: Infrastructure as a Service

- The hardware typically cheap and easily replaceable.
- Most importantly, redundancy is built into these systems
- The challenge of dealing with hardware failure is handled by the IaaS providers as data is replicated to enable recovery from failure

IaaS Benefits

- In terms of cost, these systems are able to use economies of scale to drive down the costs of individual customer storage.
- For an economically restricted organization, in order to host and share a huge number of files, this model is the only economically feasible way to share data at high availability without incurring exorbitant storage or bandwidth costs

IaaS Benefits

- IaaS solution allows us to plan for scale
  - If massive increases in storage or bandwidth occur, our solution will be able to handle it.
- Also, this model helps us avoid building infrastructure by allowing us to worry about our data rather than
  - purchasing and maintaining our own hardware,
  - hiring systems administrators, or
  - thinking about backups or electricity

Network Latency

- The average global Internet data transfer speed in 2012 was 2.3 megabits per second (Mbps), with the United States clocking in at around 5.3 Mbps
- Imagine having to transfer your 25 gigabytes of data from one place to another at a consistent speed of 5.3 Mbps.
- At this rate, the transfer will take close to 11 hours

Network Latency

- Projects like Google Fiber that aim to increase the average Internet connection toward the 1,000 Mbps range using optical fiber seem promising
  - but they may not be widespread in the United States for many years.
- Network latency will often rear its head when it comes to *big* data challenges

Data Format Use Case

- A local government installs devices that track the position and speed of each bus in the transit system every minute.
  - E.g., how well these busses stick to their planned schedules?
- As this is a civic project, the city wants to make the raw data available to people who would like to run their own analyses
  - How to structure the data so that others are easily able to make use of it?

Data Format CSV

- A common format for sharing data is comma-separated value (CSV) files.
- CSV files feature a record of data with each field in the record separated by a comma.
  - Separate records are defined by a line break.
  - While the “C” in CSV often stands for “comma,” it’s not uncommon to find formats that are delimited by other characters, such as tabs, spaces, or other, more esoteric symbols

Creating a CSV file in Python

```
import csv


my_csv_file = open('/tmp/sample.csv', 'w')
csv_writer = csv.writer(my_csv_file)
sample_row = ('Michael', 1234, 23.46, 'San Francisco,
California')

# Write a CSV row
csv_writer.writerow(sample_row)

# Result: Michael,1234,23.46,"San Francisco, California"
```

Data Format CSV Pros

- CSV is definitely a great format for "flat" data
  - data that can be represented in a single line.
  - Log data, such as that coming from Web servers and sensors, is well represented in this format.
- CSV can be fairly compact as text-based data goes
  - It's the data, pure and simple, with little markup or structure to get in the way

Data Format CSV Pros

- It is easy for most people to use CSV, as it can be
  - imported into spreadsheets,
  - ingested into databases, and
  - easily parsed programmatically.
- For logs or records that don't require data modeling beyond flat rows, CSV can be extremely useful

Data Format CSV Pros

- CSV is an excellent format for *sequential access* of data.
  - It is simple for a program to grab 1, 2, or 1,000 rows at a time from the middle of a file and just start processing.
  - Helpful in a distributed processing system for breaking up large programming tasks into many smaller ones.
  - A huge CSV file that is overwhelming the memory of a single machine? Just split it up and process the fragments

Data Format CSV Cons

- Lacks standardization
  - Little regularity in how developers create CSV output
  - E.g., some people add a few header lines, use peculiar delimiters between fields, or escape strings in eccentric ways
- Lack metadata
  - requiring those using them for sharing data to provide additional information about the file somewhere else

Data Format CSV Cons

- Bad at describing data that does not fit well into discreet rows.
  - In practice, real world data often has many dimensions, and not all of these dimensions necessarily fit into the rigid structure of CSV's rectangular regularity
- CSV files are not necessarily human readable
  - A collection of numerical values will be difficult to understand at a glance

Choosing CSV

- Choose CSV when your data is easily expressed in a flat row or if you want to ensure compatibility with nearly everything.
- CSV format is easy to parse and load, and it will always be easily supported by
  - spreadsheets,
  - databases, and
  - all kinds of other software applications due to its ubiquity

Choosing CSV

- However, don't expect CSV to do anything special
- It is not a format that is going to go out of its way to help you.
- If your audience is more likely to be less technical, CSV may be a good fit.
- If you expect your data to be used by software developers, they may prefer structured data formats such as JSON

JSON: The Programmer's Choice

- JSON (JavaScript Object Notation) is a data interchange specification that has gained much popularity with developers
- A JSON object is valid JavaScript
  - an easy data format to use for JavaScript applications.
  - However, JSON is not just for JavaScript, as parsers exist for many major programming languages.

JSON Syntax

- JSON syntax is very simple
- A file in JSON can potentially be faster to parse than a similar document encoded in XML due to a lighter-weight syntax
- Placing a collection of newline-delimited JSON objects together in a file makes the format easy to parse programmatically, in the same way that CSV provides sequential access to data

Comparison of CSV and JSON

- # CSV example

first_name,last_name,book,date

"Michael", "Manoochehri", "Data Just Right",2013

- // JSON example

```
{
"name": "Michael",
"book":{"title":"Data Just Right","date":"2013"}
}
```

Choosing JSON

- Hosting a collection of data in newline-delimited JSON format is a good start for data exchange when coders are involved, and this format will make application developers and non-relational database administrators happy
- JSON is not a great format for a general population that expects to be able to load files directly into a spreadsheet (for that application, use CSV)

Character Encoding

- Character encoding is the process by which a computer represents all the characters and symbols necessary for exchanging information
- A very simple example of character encoding is Morse code, which represents letters in the alphabet as pulses that can be transmitted over a wire or visually

Character Encoding

- American Standard Code for Information Interchange (**ASCII)** works great for representing characters from the English language
- **Unicode Standard** aims to define a set of standard encodings for all the characters in the world
  - If you've got lots of data lying around that is not already encoded in UTF-8 or UTF-16, then convert it

Working with Text Files

- When working with large amounts of text data, sometimes the quickest way to solve a problem is to use some of the original Unix command-line utilities, such as split, grep,and sed.
- After many decades, this software is still in widespread use today; thanks to open-source implementations of Unix such as GNU/Linux, the utilities might even be used more than ever before

Working with Text Files – sed

- Do you have annoying header information in your CSV file? Remove the first two lines of a file and place the result in a new file:

```
sed '1,2d' file > newfile
```

- Is the data in your source file delimited by an odd character, such the caret (^)? Replace those pesky characters with commas!

```
sed 's/\^/,/g' original_file > new_file
```

Working with Text Files – split

- Another great utility for dealing with very large files is split, which, as the name implies, splits large files into smaller ones.
- For example, if you have a very large file that you need to split into chunks of 500 Mb at most while preserving the integrity of line endings (avoiding broken lines), use this command:

```
split -C 500m your_large_file.csv newfile
```

File Transformations

- In practice, converting many files from one format to another can be a very involved process.
- What if you've got a lot of data in a particular format, and you need to convert it to another format?
- This data transformation process can sometimes be daunting

File Transformations

- Transforming a large collection one document at time can take enormous amounts of time.
- Fortunately, it is also a task well suited to distributed systems.
- When confronted with hundreds and even thousands of separate documents, it's helpful to run these tasks in parallel using a large number of distributed resources

File Transformations

- The most popular open-source distributed computing framework is **Hadoop**.
- Hadoop is an open-source implementation of the MapReduce framework, which allows data processing tasks to be split across a large number of separate machines
  - Hadoop was originally inspired by Google's MapReduce research paper

Data in Motion: Data Serialization Formats

- XML, JSON, and even plain old CSV files can all be considered members of a class of objects used in the process of converting data into bits that a computer can understand and move from one place to another.
- This process is known as **data serialization**

Data Serialization

- Data systems often make use of many machines across a network
  - Some machines specialize in collecting data from a large number of inputs quickly.
  - Others are tasked with running batch processes that analyze the data.
- When building systems that move data from one system to another, sending the output of one process to the input of another is a common task

Data Serialization

- The network is always going to be the slowest step in this transferring process.
- To help provide the most efficient data transfer, it's beneficial to represent data in the most compact way possible
- Although it's possible to compress files in data formats before sending them off, the developer still needs to handle the compression and decompression steps himself

Data Serialization

- A naïve approach to this problem would be to convert data into some type of byte array
  - a binary representation of the data.
- This approach might reduce the size of the data, but, unfortunately, each system would have to know beforehand exactly how the data was serialized so that it could later be deserialized

Data Serialization

- Hard coding the encoding and decoding functions into the application would work, but it would be problematic if anything about the data model changed.
- If an application pipeline depended on multiple systems built with different programming languages, the work involved in changing these functions would be considerable

Some Solution to Serialization

- Provide a description of the data, or schema, and define it somewhere that is common to both sender and receiver
- Build a standard interface with programming language support to serialize the data into both the message sender and receiver

Apache Thrift

- Apache Thrift is an open-source project originally developed at Facebook to provide a generic data solution for data serialization.
  - allow developers to define a configuration file that describes the data that will be serialized.
  - A code generator is run to produce a server that handles the data serialization in the language specified

Protocol Buffers

- Google's Protocol Buffers are very similar to Thrift, and the software to compile
  - Protocol Buffers is also available to use as open-source software
- Like Thrift, Protocol Buffers are used in a variety of projects.
- While PB is generally used for moving data around, some organizations are using it for persistent files

Apache Avro

- Apache Avro is a relatively new data serialization format that combines some of the best features of all the technologies
  - Unlike Apache Thrift or Protocol Buffers, Avro data is self-describing and uses a JSON-schema description to describe each individual field
  - Avro also natively supports compression so that developers don’t have to spend much time worrying about building this logic themselves

Summary

Summary

- Hadoop and many analytics tools are available in the open source community
- We covered different strategies to process Big Data
- Visualization is critical in Big Data to enable extracting value from information
- Choosing a file format for your hosted files can be a challenge, but not if you know how your audience may use your data