DATA7001
INTRODUCTION TO DATA SCIENCE

Module 2 Getting the Data I Need

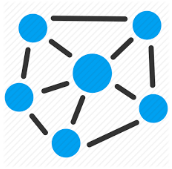
Module Topics

- Types of Data
- Data Ingestion
- Managing Data Privacy
- Sampling Big Data

Types of Data

- Structured
- Text
- Spatial
- Time Series
- Graph
- Multimedia

Name	Age	Course	GPA
xyz	25	D01	5.0
abc	22	D01	3.5
jlk	24	D01	6.5



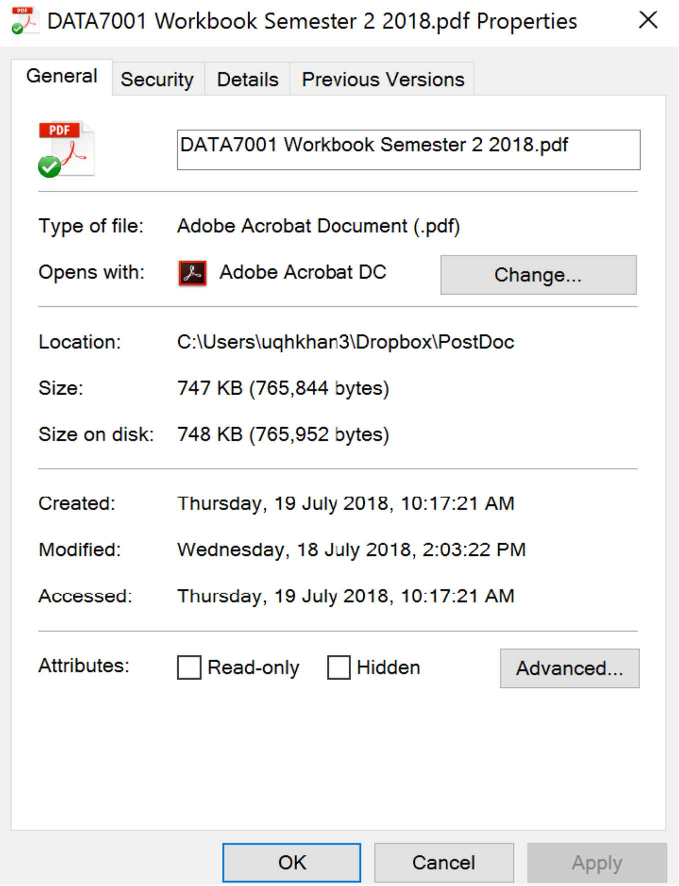
- …and Meta-data

What is Meta Data?

Data

DATA7001 Workbook Semester 2 2018

Meta Data



DATA7001 Workbook Semester 2 2018.pdf Properties

DATA7001 Workbook Semester 2 2018.pdf

Type of file: Adobe Acrobat Document (.pdf)

Opens with: Adobe Acrobat DC

Location: C:\Users\uqhkhan3\Dropbox\PostDoc

Size: 747 KB (765,844 bytes)

Size on disk: 748 KB (765,952 bytes)

Created: Thursday, 19 July 2018, 10:17:21 AM

Modified: Wednesday, 18 July 2018, 2:03:22 PM

Accessed: Thursday, 19 July 2018, 10:17:21 AM

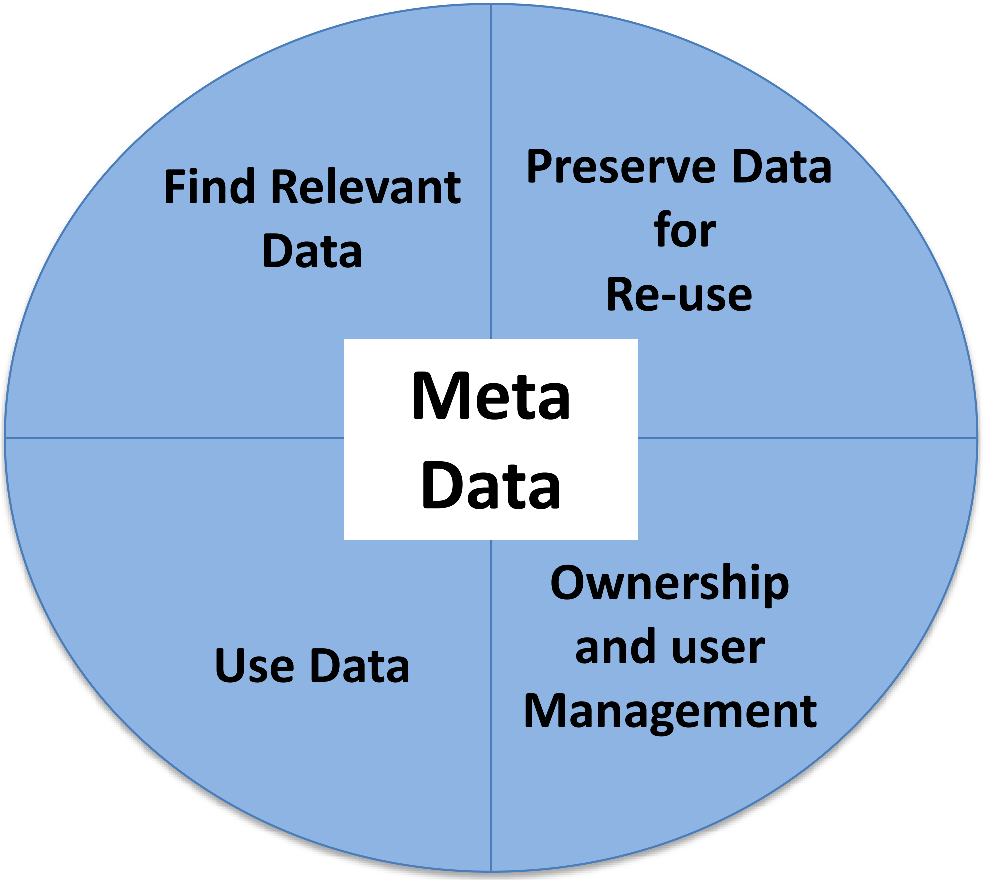
Attributes: □ Read-only □ Hidden

Meta-data is data about other data

Types of Meta-data

Type	Description	Use
Descriptive	• Describes the data, such as author, title, abstract, key terms • Difficult to create and manage	Enable searching and retrieval of data
Administrative	• Technical data on creation and quality control such as rights management, access control and use requirements • Created and managed by automation and by procedure.	Facilitates management of data for use and re-use
Structural	• Describes the information hierarchy of data, e.g., table of contents, frame index, versioning info. • Created when the data is created	Facilitates information navigation. Provides a higher view of how data is put together in a meaningful way.

Why Meta-data?



Find Relevant Data

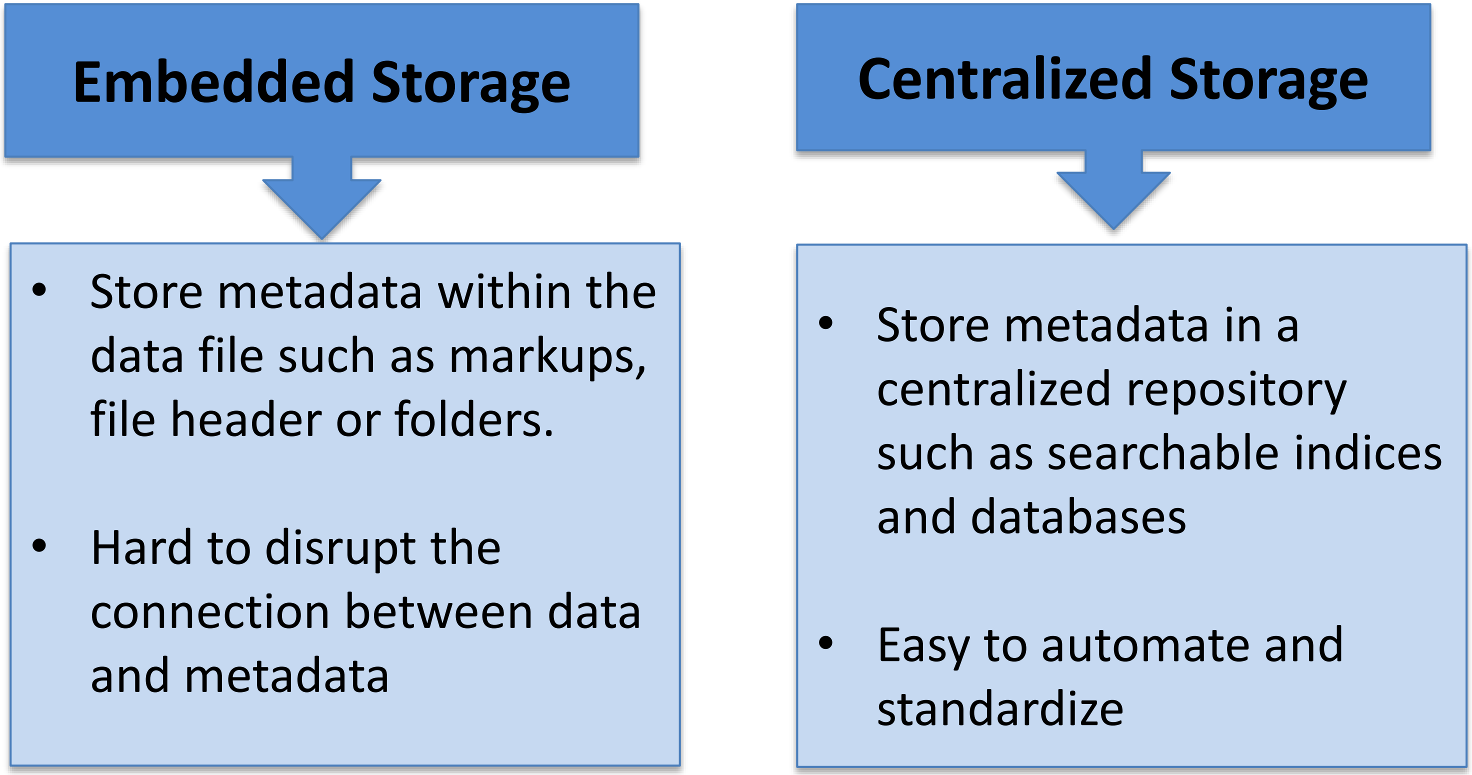
Preserve Data for Re-use

Meta Data

Use Data

Ownership and user Management

Where is meta-data ?



Embedded Storage

- Store metadata within the data file such as markups, file header or folders.
- Hard to disrupt the connection between data and metadata

Centralized Storage

- Store metadata in a centralized repository such as searchable indices and databases
- Easy to automate and standardize

Metadata format is routinely standardised, and for good reason. If the metadata cannot be understood, then the data cannot be understood....

Meta-data standards

Aspects of meta-data standards

Syntax - the layout, character encoding, punctuation of the metadata. XML is a good example.

Semantics - the terminology, interpretation and operationalisation of metadata. This includes dictionaries of tags, keywords, terms, classes in classification systems, and permitted ways for acting on each of these - such as how to render a tag, decrypt a codec, or route an address field.

Popular standards

There is a whole forest of metadata standards! The introduction of XML has helped standardise syntax in some areas, but the semantics of different disciplines are very different, and no two people interpret data in exactly the same way.

Semantic metadata standards :

- Library of Congress Classification,
- DDI for social science data
- TEI for archived text
- Dublin core for internet content
- MPEG for media
- RDF for relationships

More on Meta Data

Further reading on meta-data

The Wikipedia page on “metadata” is fine as a starting point. Metadata standards change frequently, so using this page as a jumping off point to read about ongoing standardisation projects is actually a good idea.

For more theory on library and information science, try:

Robert Colomb (2002). *Information Spaces, The Architecture of Cyberspace*. Springer London.

Gerard Salton (1989). *Automatic Text Processing*. Addison-Wesley Publishing.

Types of Data

- Structured
- Text
- Spatial
- Time Series
- Graph
- Multimedia

Structured (relational)

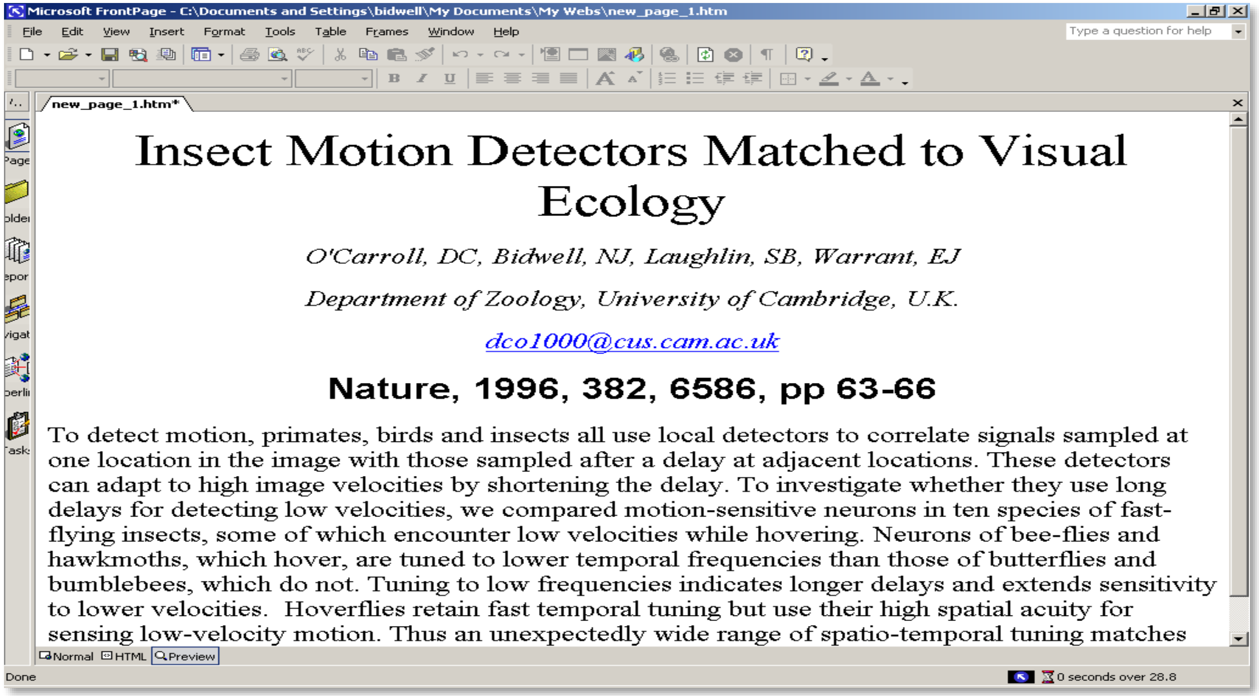
Title	Journal	Vol	Issue	Date	Pages
Insect Motion Detectors Matched to Visual Ecology	Nature	382	6586	1999	Pp 63-66
Information systems success: The quest for the dependent variable	Information Systems Research	3	1	NULL	60-95

Semi-structured (xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www. http://www.itee.uq.edu.au/~nbidwell/schemas/paper”
          <xsd:complexType name=“publishType">
                  <xsd:sequence>
                  <xsd:element ref=“journal"/>
                  <xsd:element name=“date" type="xsd:date“ minOccurs=“1"/>
                  <xsd:element name=“volume" type="xsd:string"/>
                  <xsd:element name=“issue" type="xsd:string"/>
                  <xsd:element name=“page" type="xsd:string"/>
                  </xsd:sequence>
          </xsd:complexType>

</xsd:schema>
```

Unstructured (document)



Microsoft FrontPage - C:\Documents and Settings\bidwell\My Documents\My Webs\new_page_1.htm
File Edit View Insert Format Tools Table Frames Window Help
Type a question for help
new_page_1.htm*
Page
older
epor
vigat
perli
Task
Normal
HTML
Preview
Done
0 seconds over 28.8

Insect Motion Detectors Matched to Visual Ecology

O'Carroll, DC, Bidwell, NJ, Laughlin, SB, Warrant, EJ

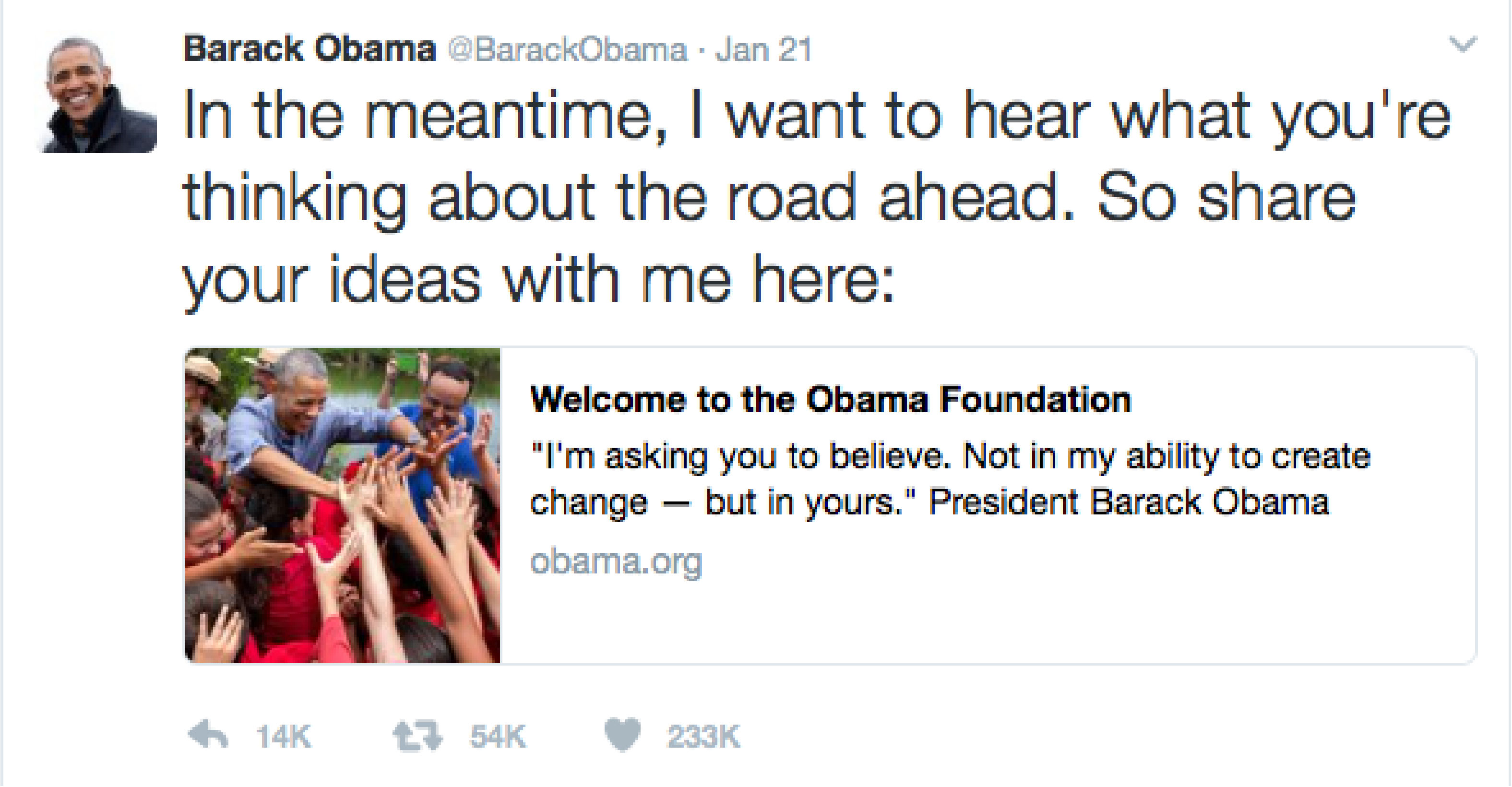
Department of Zoology, University of Cambridge, U.K.

dco1000@cus.cam.ac.uk

Nature, 1996, 382, 6586, pp 63-66

To detect motion, primates, birds and insects all use local detectors to correlate signals sampled at one location in the image with those sampled after a delay at adjacent locations. These detectors can adapt to high image velocities by shortening the delay. To investigate whether they use long delays for detecting low velocities, we compared motion-sensitive neurons in ten species of fast-flying insects, some of which encounter low velocities while hovering. Neurons of bee-flies and hawkmoths, which hover, are tuned to lower temporal frequencies than those of butterflies and bumblebees, which do not. Tuning to low frequencies indicates longer delays and extends sensitivity to lower velocities. Hoverflies retain fast temporal tuning but use their high spatial acuity for sensing low-velocity motion. Thus an unexpectedly wide range of spatio-temporal tuning matches

Unstructured (short text)



Barack Obama @BarackObama · Jan 21

In the meantime, I want to hear what you're thinking about the road ahead. So share your ideas with me here:

Welcome to the Obama Foundation

"I'm asking you to believe. Not in my ability to create change — but in yours." President Barack Obama

obama.org

14K 54K 233K

Spatial

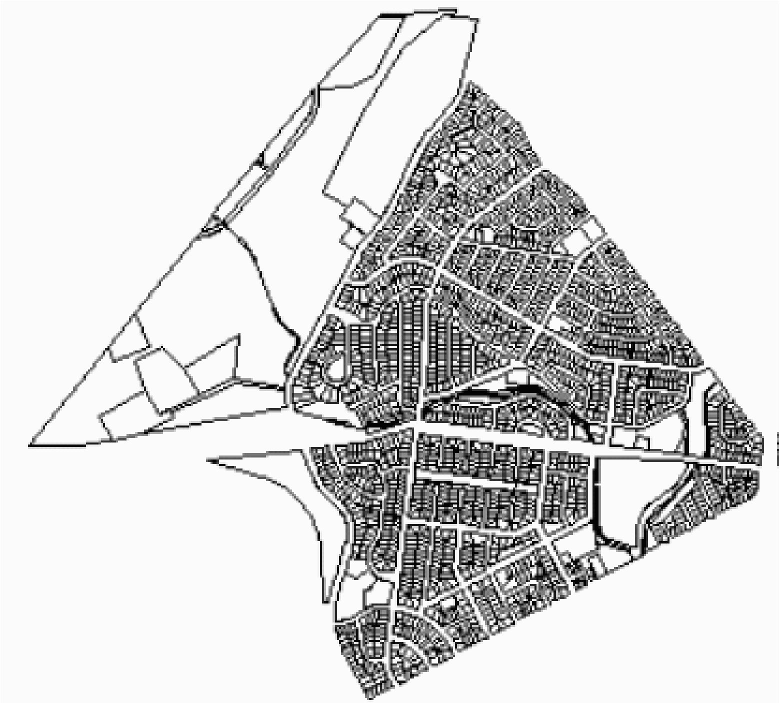
Any data with a location component

- Base-map data
  - Control points, topographic contours, building sites
- Natural area data
  - Soil types, landuse (industrial, agriculture, zoning etc), vegetation, water (rivers, ponds etc)
- Manmade area data
  - School districts, emergency service areas
- Land records data
  - Lot boundaries, zoning, easements
- Network data
  - Utilities (phones, sewers, water, electricity etc)
  - Roads (centerlines, curb lines, intersections, lights)

Spatial data representations

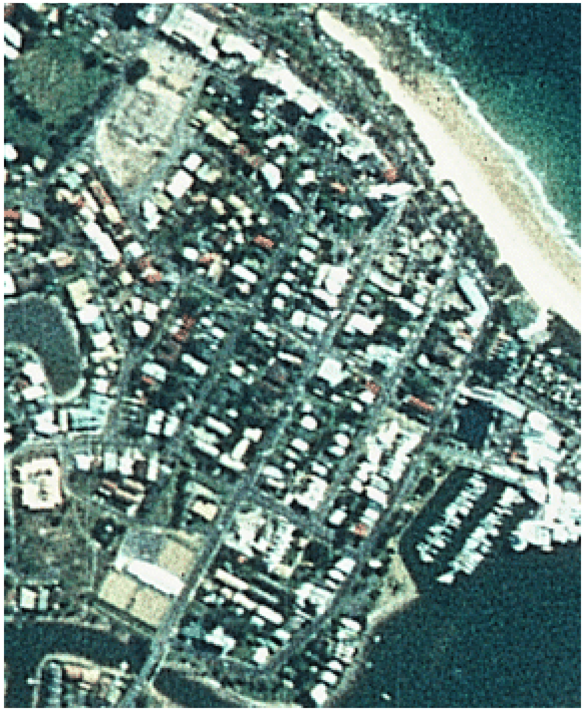
Vector Data

- A Canberra Suburb



Raster Data

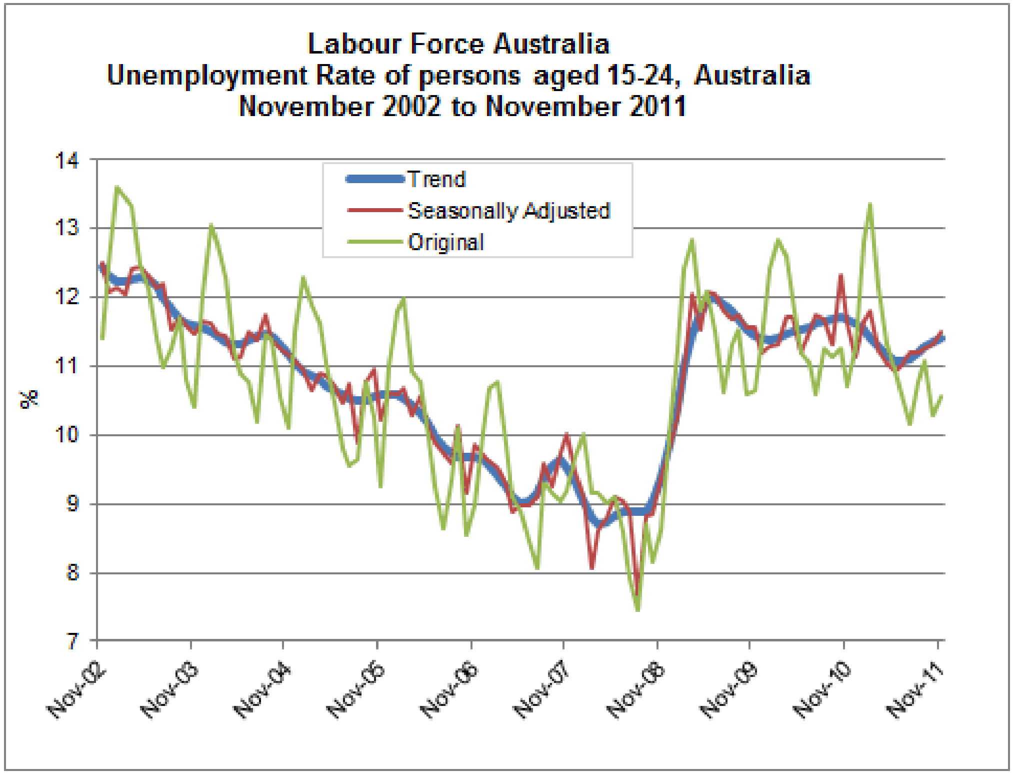
- Maroochy, Sunshine Coast



Time Series

A series of measurements taken at successive regularly spaced time intervals

“For example, measuring the level of unemployment each month of the year would comprise a time series. This is because employment and unemployment are well defined, and consistently measured at equally spaced intervals.” [Australian Bureau of Statistics]



Types of Time Series Data

- A **stock series** is a measure of certain attributes at a point in time and can be thought of as “stock takes”. For example, the annual ABS *Prisoners in Australia* collection is a stock measure because it is a count of the number of persons in custody who were the legal responsibility of adult corrective services agencies on the night of 30 June each year.

- A **flow series** is a series which is a measure of activity over a given period. For example, the quarterly ABS *Corrective Services, Australia* collection is a flow measure as it provides prisoner counts taken on each day of the month which are summed and divided by the number of days in that month to determine the mean (average) daily prisoner number for that month

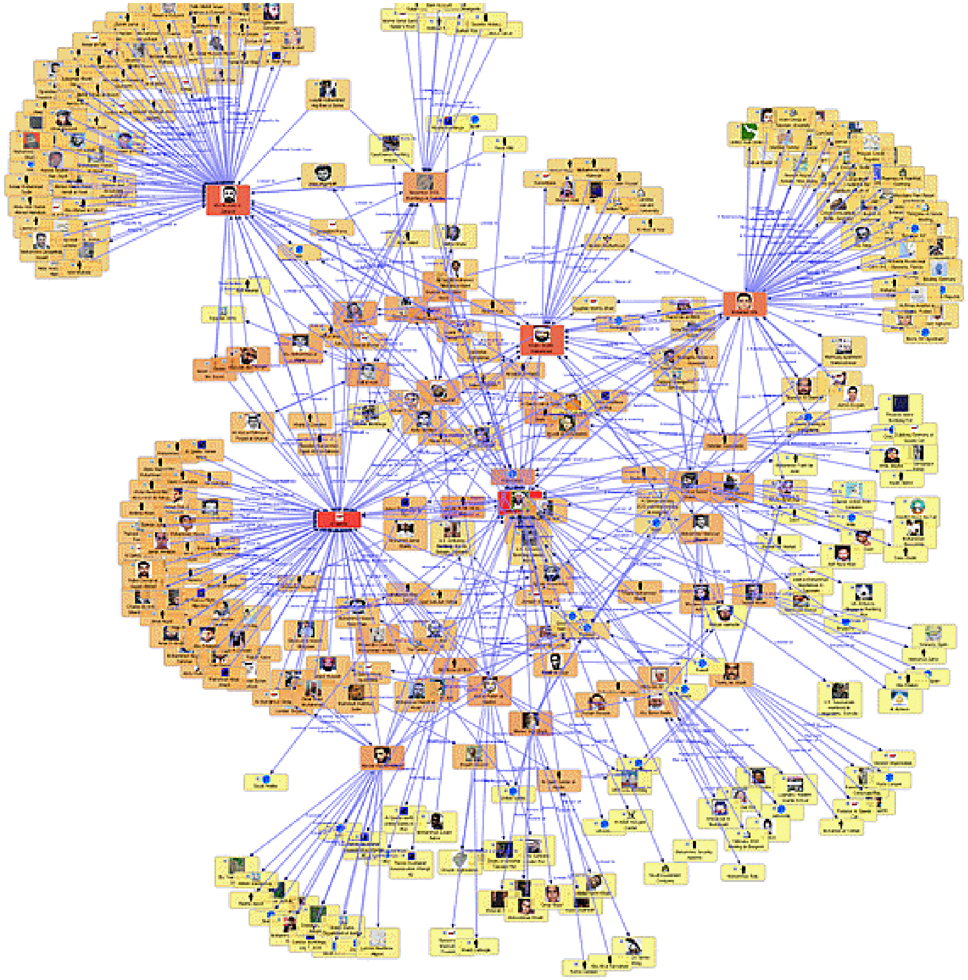
Graph Data

Widely used data structure

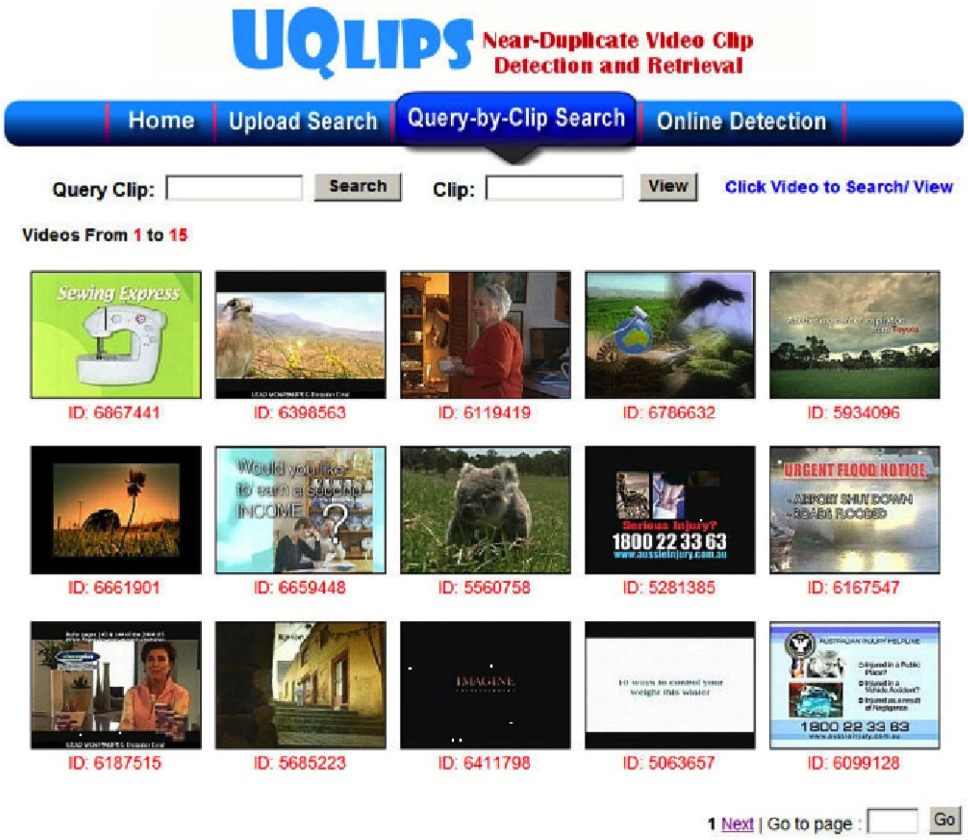
- Nodes: entities such as people, websites, objects
- Edges: relationships between nodes, e.g 'friend of'
- Properties: attributes of nodes such as age group

Storage

- **Relational** (table) known to have flexibility and scalability limitations
- **Key-value** stores (~NoSQL databases e.g Neo4j) but also known to have issues of integrity and adoption



Multimedia Data



Multimedia data includes multiple data types including: text, images, graphics, audio and video.

Task and Discussion

Given two examples of *interesting* analytics that will need two or more different types of data

Example1. Sentiment analysis on product reviews social data (short text) displayed on a map (spatial data) with visualizations to support population demographics obtained from open census data (structured).

Example 2.

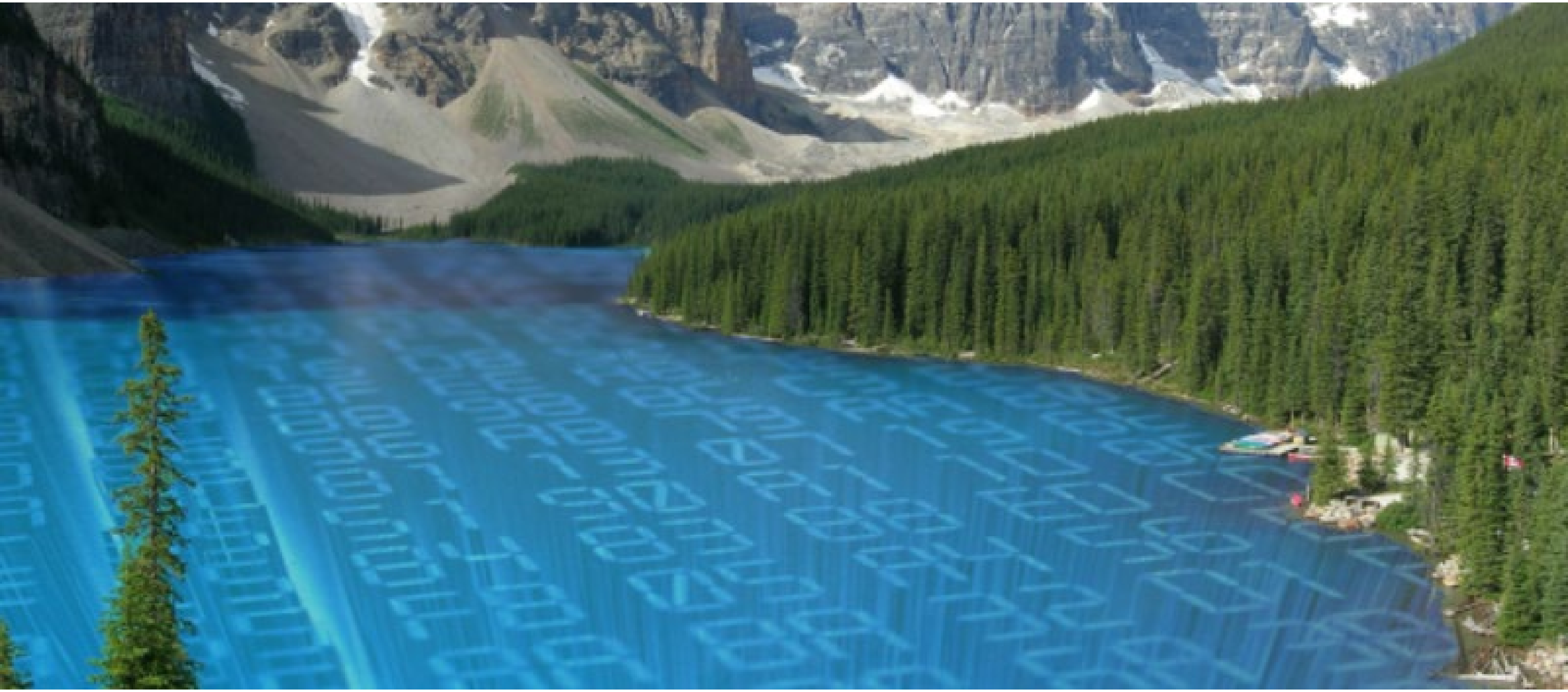
Example 3.

Module Topics

- Types of Data
- **Data Ingestion**
- Managing Data Privacy
- Sampling Big Data

- What data do you need?
  - **Why** do you need it?
  - Do you need **all of it**, or a **sample** will do?
- How do you get the data?
  - Do you own it, or will you beg, buy, or scrape it?
  - How fast is your data arriving (stream, batch, or snap)?
  - Are you authorized to acquire that data?
- Where do you keep the data?
  - In what form are you going to store (*ingest*) the data?
  - For how long do you need to keep the data?

Data Lake



A data lake is a storage repository that holds a vast amount of raw data in its native format until it is needed. While a hierarchical data warehouse stores data in files or folders, a data lake uses a flat architecture to store data.

Recall from Module 1 …

GET ALL THE INFORMATION YOU CAN, WE'LL THINK OF A USE FOR IT LATER.

Dumping everything into e.g a Hadoop Distributed File System (HDFS), with a plan to do something interesting with it some day is going to turn your **DATA LAKE** into a **DATA SWAMP**

https://theconversation.com/use-it-or-lose-it-the-search-for-enlightenment-in-dark-data-68167

Automating Data Ingestion

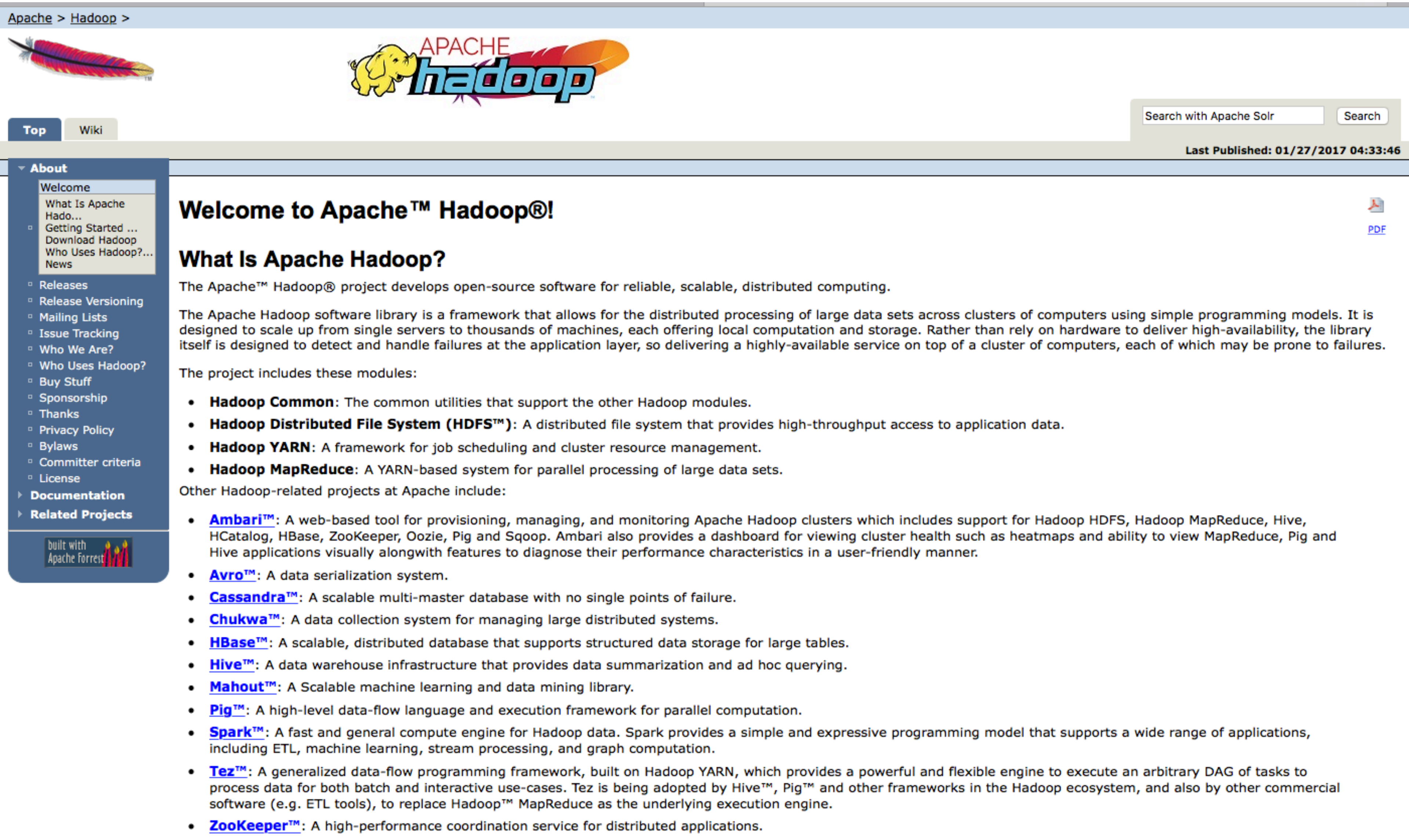
Challenges

- Need for customized scripts for each (multiple) source of data
- Huge demands on power, computing and bandwidth for high volume data (streams)
- Data loss due to outages from unstable connectivity
- Lack of security and difficulties in managing data access

- Data ingestion (typically) refers to HDFS (Hadoop Distributed File System)
- As the **Hadoop** ecosystem matures, many tools are available to simplify data ingestion e.g hortonworks and cloudera
- New frameworks have emerged to overcome some of the limitations of Hadoop e.g **Spark**

→ Accelerate the time to (start of) analytics

http://hadoop.apache.org



Welcome to Apache™ Hadoop®!

What Is Apache Hadoop?

The Apache™ Hadoop® project develops open-source software for reliable, scalable, distributed computing.

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

The project includes these modules:

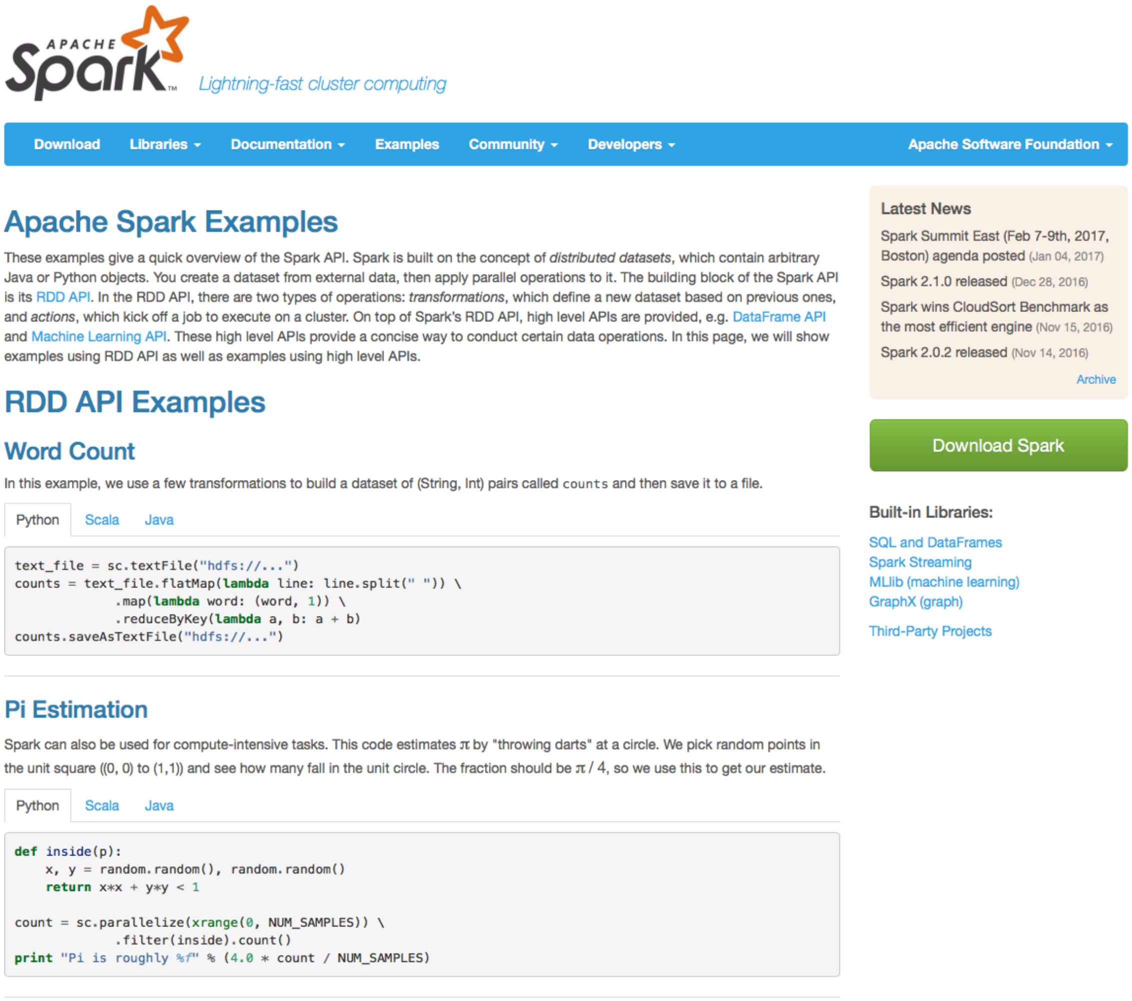
- **Hadoop Common**: The common utilities that support the other Hadoop modules.
- **Hadoop Distributed File System (HDFS™)**: A distributed file system that provides high-throughput access to application data.
- **Hadoop YARN**: A framework for job scheduling and cluster resource management.
- **Hadoop MapReduce**: A YARN-based system for parallel processing of large data sets.

Other Hadoop-related projects at Apache include:

- **Ambari™**: A web-based tool for provisioning, managing, and monitoring Apache Hadoop clusters which includes support for Hadoop HDFS, Hadoop MapReduce, Hive, HCatalog, HBase, ZooKeeper, Oozie, Pig and Sqoop. Ambari also provides a dashboard for viewing cluster health such as heatmaps and ability to view MapReduce, Pig and Hive applications visually alongwith features to diagnose their performance characteristics in a user-friendly manner.
- **Avro™**: A data serialization system.
- **Cassandra™**: A scalable multi-master database with no single points of failure.
- **Chukwa™**: A data collection system for managing large distributed systems.
- **HBase™**: A scalable, distributed database that supports structured data storage for large tables.
- **Hive™**: A data warehouse infrastructure that provides data summarization and ad hoc querying.
- **Mahout™**: A Scalable machine learning and data mining library.
- **Pig™**: A high-level data-flow language and execution framework for parallel computation.
- **Spark™**: A fast and general compute engine for Hadoop data. Spark provides a simple and expressive programming model that supports a wide range of applications, including ETL, machine learning, stream processing, and graph computation.
- **Tez™**: A generalized data-flow programming framework, built on Hadoop YARN, which provides a powerful and flexible engine to execute an arbitrary DAG of tasks to process data for both batch and interactive use-cases. Tez is being adopted by Hive™, Pig™ and other frameworks in the Hadoop ecosystem, and also by other commercial software (e.g. ETL tools), to replace Hadoop™ MapReduce as the underlying execution engine.
- **ZooKeeper™**: A high-performance coordination service for distributed applications.

http://spark.apache.org

Apache Spark is an open-source cluster-computing framework. Originally developed at the University of California, Berkeley's AMPLab, the Spark codebase was later donated to the Apache Software Foundation, which has maintained it since. Spark provides an interface for programming entire clusters with implicit data parallelism and fault-tolerance. [wikipedia]



Apache Spark Examples

These examples give a quick overview of the Spark API. Spark is built on the concept of *distributed datasets*, which contain arbitrary Java or Python objects. You create a dataset from external data, then apply parallel operations to it. The building block of the Spark API is its RDD API. In the RDD API, there are two types of operations: *transformations*, which define a new dataset based on previous ones, and *actions*, which kick off a job to execute on a cluster. On top of Spark's RDD API, high level APIs are provided, e.g. DataFrame API and Machine Learning API. These high level APIs provide a concise way to conduct certain data operations. In this page, we will show examples using RDD API as well as examples using high level APIs.

RDD API Examples

Word Count

In this example, we use a few transformations to build a dataset of (String, Int) pairs called `counts` and then save it to a file.

```
text_file = sc.textFile("hdfs://...")
counts = text_file.flatMap(lambda line: line.split(" ")) \
             .map(lambda word: (word, 1)) \
             .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile("hdfs://...")
```

Pi Estimation

Spark can also be used for compute-intensive tasks. This code estimates π by "throwing darts" at a circle. We pick random points in the unit square ((0, 0) to (1,1)) and see how many fall in the unit circle. The fraction should be π / 4, so we use this to get our estimate.

```
def inside(p):
    x, y = random.random(), random.random()
    return x*x + y*y < 1

count = sc.parallelize(xrange(0, NUM_SAMPLES)) \
             .filter(inside).count()
print "Pi is roughly %f" % (4.0 * count / NUM_SAMPLES)
```

Latest News

Spark Summit East (Feb 7-9th, 2017, Boston) agenda posted (Jan 04, 2017)

Spark 2.1.0 released (Dec 28, 2016)

Spark wins CloudSort Benchmark as the most efficient engine (Nov 15, 2016)

Spark 2.0.2 released (Nov 14, 2016)

Archive

Built-in Libraries:

- SQL and DataFrames
- Spark Streaming
- MLlib (machine learning)
- GraphX (graph)

Third-Party Projects

Examples of Data Ingestion

- CSV to MySQL (relational data)
- TXT to HDFS (hadoop distributed file system)

- Checkout tools for automating ingestion and other housekeeping tasks
  - Hortonworks
  - Cloudera

POLL QUESTIONS – DATA TYPES

Module Topics

- Types of Data
- Data Ingestion
- **Managing Data Privacy (next part)**
- Sampling Big Data