UNIVERSITY OF TWENTE.

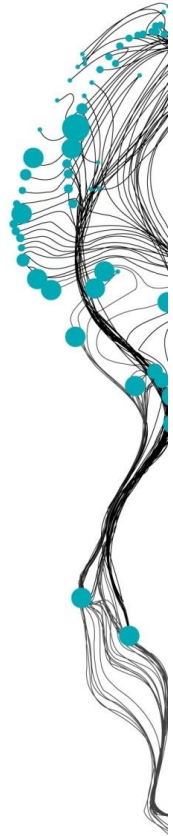
BIG DATA SOLUTIONS DISTRIBUTED COMPUTING

HADOOP, SPARK AND DASK

Presented by: Mahdi Farnaghi

Developed by: Raúl ZURITA-MILLA (2021),
Updated by: Mahdi Farnaghi (2022)

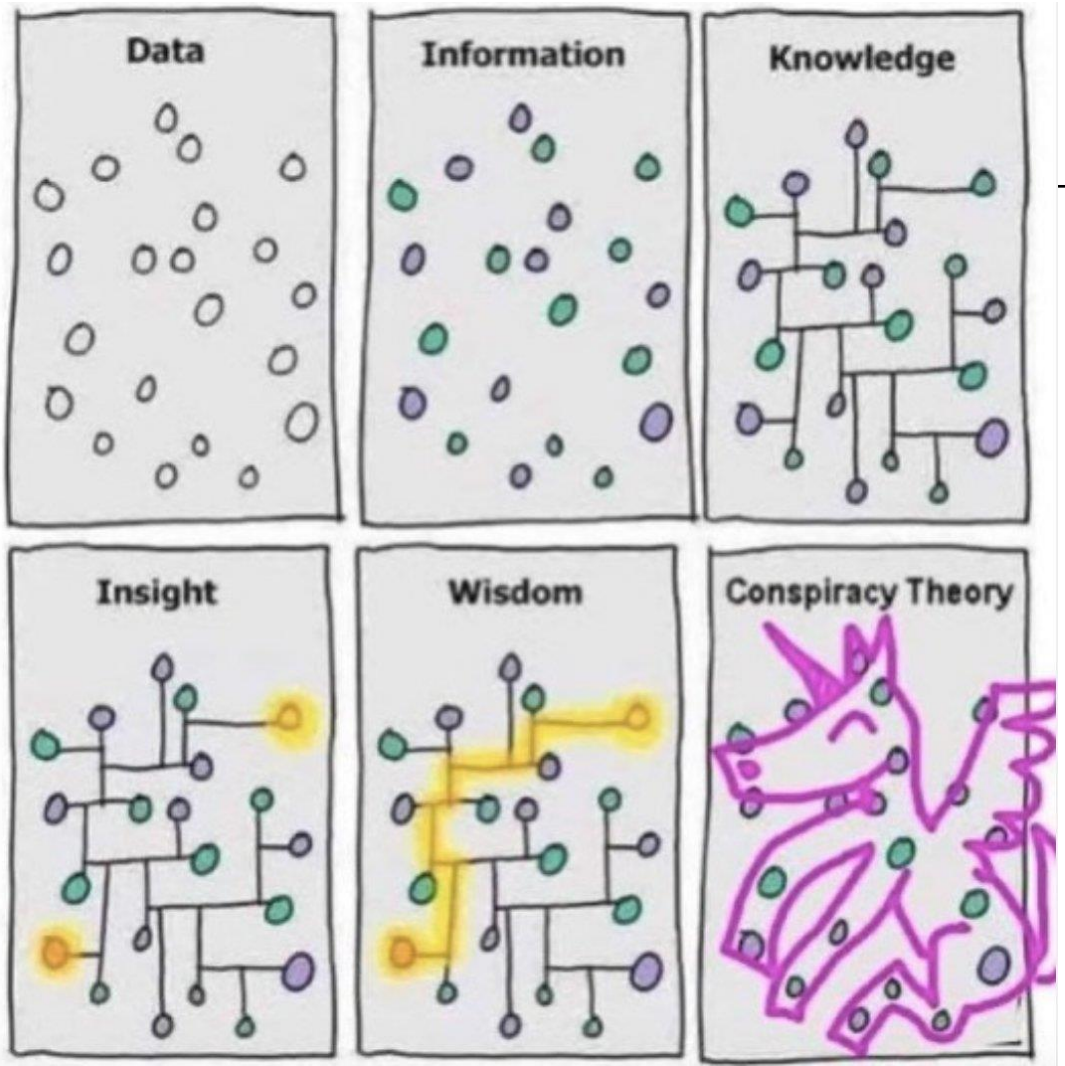
ITC FACULTY OF GEO-INFORMATION SCIENCE AND EARTH OBSERVATION



BIG DATA (SCIENCE)

- Raw data fundamentally has no value

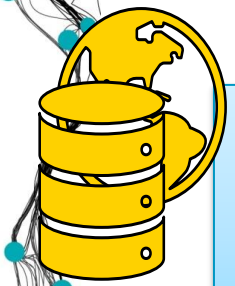
- Big geodata processing →
  - Better decisions, insights and information for all
  - (managers, decision makers, scientists, citizens, etc.)



ITC

UNIVERSITY OF

BIG GEO DATA CHALLENGES



Tackle real-world problems

Large Study Area

Conventional computational solutions do not work

Computing solutions

Workstation (CPU, GPU, TPU)

- Store data
- Develop a code or use a software
- Load the data into the memory
- Run
- Generate a result

Distributed computing (Cluster of machines working together)

- Distributed storage
- Distributed processing
- Develop a code
- The code run on the nodes where data resides
- The results collected and aggregated

OPEN-SOURCE DISTRIBUTED COMPUTING SOLUTIONS THAT SUPPORT BIG DATA

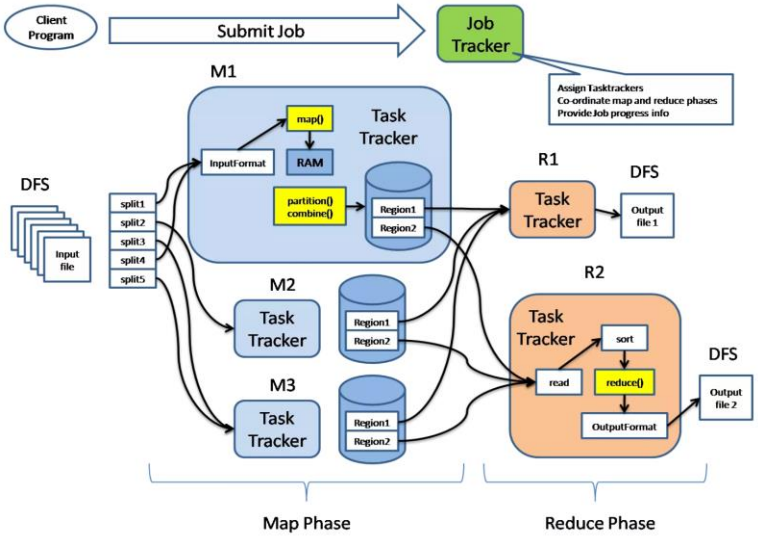
THREE DISTRIBUTED COMPUTING SOLUTIONS

HADOOP

- Hadoop is a collection of open-source software utilities that facilitates the use of networked computers to work with big data.

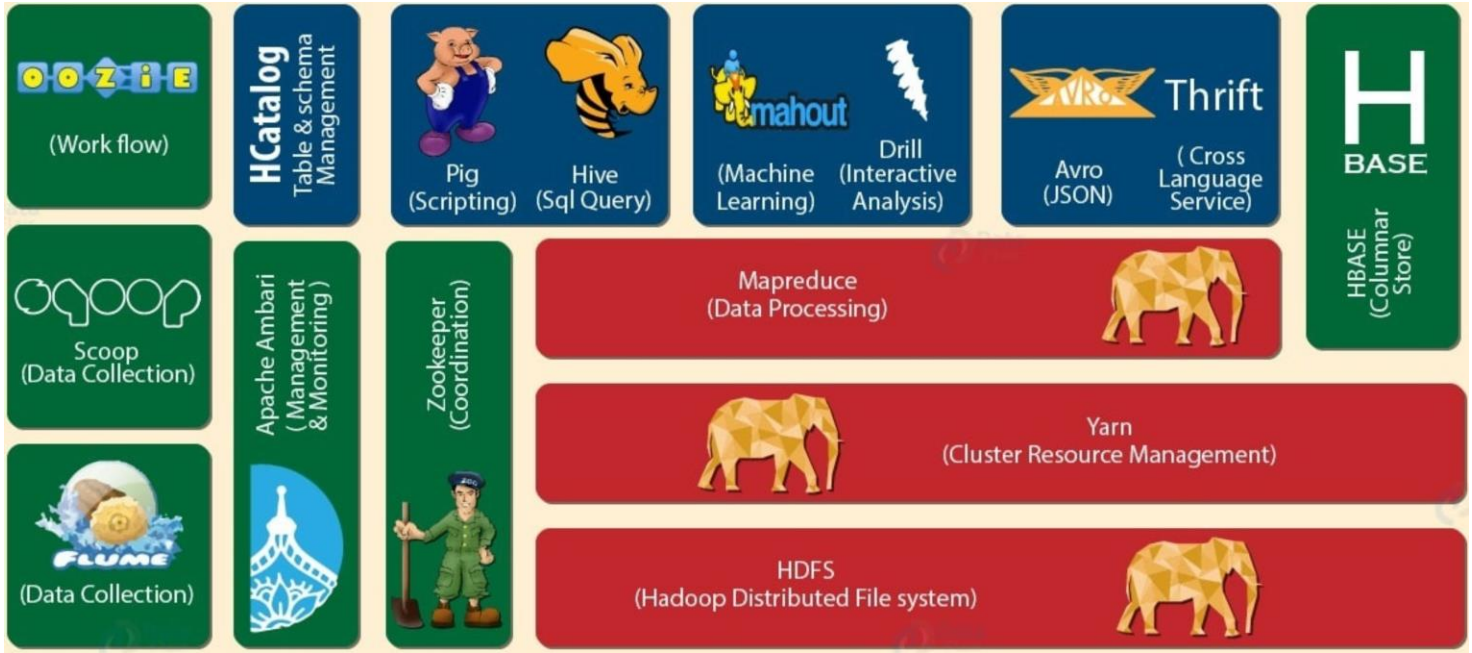
- Eliminates the bottlenecks of client-server models.

Developer(s)	Apache Software Foundation
Initial release	April 1, 2006; 13 years ago[1]



https://www.datadriveninvestor.com/2018/06/08/an-introduction-to-hadoop-for-noobs/

HADOOP ECOSYSTEM



https://www.quora.com/What-is-a-Hadoop-ecosystem

HADOOP – HDFS AND MAP REDUCE

Hadoop relies on two main pillars:

- HDFS is a java-based system used to handle big data. HDFS creates multiple replicas of data blocks and distributes them on compute nodes in a cluster, which make it easier and faster to find files.
- MapReduce is designed to process large data sets across a cluster of computers. It has two parts: the Mapper and the Reducer.

WHY HADOOP?

- HDFS is highly scalable (thousands of nodes and PBs of data).
- It distributes large data sets across clusters of commodity computers (cheap and widely available)
- It is good at handling unstructured data (a common data type).
- It replicates data across computers (if one fails, data processing can still happen by using data from another one). Fault tolerance.
- Open-Source

WHY NOT HADOOP?

- Do you really have big data?

  Up to 5TB buy an ext. HDD and just use Postgres!

Hadoop << SQL, Python Scripts

- In terms of expressing your computations, Hadoop is strictly inferior to SQL. There is no computation you can write in Hadoop that you cannot write more easily in either SQL, or Python (for unstructured data).

https://www.chrisstucchio.com/blog/2013/hadoop_hatred.html

WHY NOT HADOOP?

Hadoop's star dims in the era of cloud object data storage and stream computing

ANALYSIS BY JAMES KOBIELUS

One of the most noteworthy findings from Wikibon's annual update to our big data market forecast was how seldom Hadoop was mentioned in vendors' roadmaps.

I wouldn't say that Hadoop — open-source software for storing data and running applications on large hardware clusters — is entirely dead. Most big-data analytics platform and cloud providers still support such Hadoop pillars as YARN, Pig, Hive, HBase, ZooKeeper and Ambari.

However, none of those really represents the core of this open-source platform in the way that the Hadoop Distributed File System or HDFS does. And HDFS is increasingly missing from big data analytics vendors' core platform strategies.

https://siliconangle.com/2018/07/09/hadoops-star-dims-era-cloud-object-data-storage-stream-computing/

HADOOP-BASED SOLUTIONS

http://spatialhadoop.cs.umn.edu/

SpatialHadoop

A MapReduce Framework for Spatial Data

DATASETS PUBLICATIONS TUTORIALS DOWNLOAD

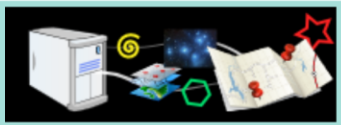
Analyze your spatial data efficiently

SpatialHadoop is an open source MapReduce extension designed specifically to handle huge datasets of spatial data on Apache Hadoop. SpatialHadoop is shipped with built-in spatial high level language, spatial data types, spatial indexes and efficient spatial operations.

Spatial language

Interact with the system easily with a simple high level language. More»

Spatial data types



Load all your datasets in SpatialHadoop with the built-in spatial data types

Spatial indexes

Store your data efficiently in a spatial index of your choice (Grid file, R-tree or R+-tree)

Spatial operations

Start analyzing your data on large clusters with built-in spatial operations

ITC

HADOOP-BASED SOLUTIONS

GEO-SPATIAL INFORMATION SCIENCE, 2018
VOL. 21, NO. 2, 102–114
https://doi.org/10.1080/10095020.2017.1413798

A spatiotemporal algebra in Hadoop for moving objects

Mohamed S. Bakli[a], Mahmoud A. Sakr[b,c] and Taysir Hassan A. Soliman[a]

[a]Faculty of Computers and Information, Assiut University, Assiut, Egypt; [b]Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt; [c]Université libre de Bruxelles, Bruxelles, Belgium

ABSTRACT

Spatiotemporal data represent the real-world objects that move in geographic space over time. The enormous numbers of mobile sensors and location tracking devices continuously produce massive amounts of such data. This leads to the need for scalable spatiotemporal data management systems. Such systems shall be capable of representing spatiotemporal data in persistent storage and in memory. They shall also provide a range of query processing operators that may scale out in a cloud setting. Currently, very few researches have been conducted to meet this requirement. This paper proposes a Hadoop extension with a spatiotemporal algebra. The algebra consists of moving object types added as Hadoop native types, and operators on top of them. The Hadoop file system has been extended to support parameter passing for files that contain spatiotemporal data, and for operators that can be unary or binary. Both the types and operators are accessible for the MapReduce jobs. Such an extension allows users to write Hadoop programs that can perform spatiotemporal analysis. Certain queries may call more than one operator for different jobs and keep these operators running in parallel. This paper describes the design and implementation of this algebra, and evaluates it using a benchmark that is specific to moving object databases.;

ARTICLE HISTORY
Received 31 December 2016
Accepted 12 November 2017

KEYWORDS
Spatiotemporal algebra; Hadoop; MapReduce; moving objects

SPARK

- Spark is another open-source distributed framework for cluster computing.
- Spark provides faster and easier to use analytics than Hadoop.

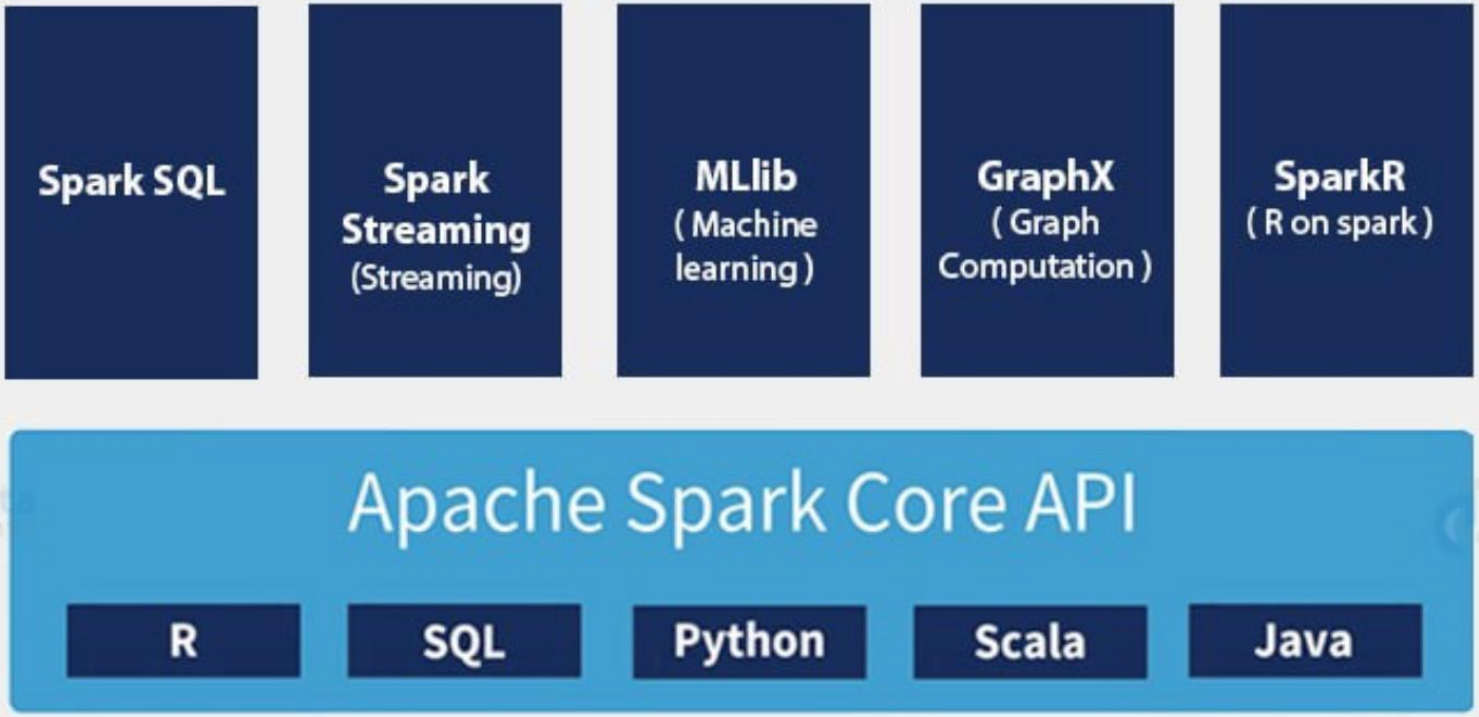
Original author(s)	Matei Zaharia
Developer(s)	Apache Software Foundation, UC Berkeley AMPLab, Databricks
Initial release	May 26, 2014; 5 years ago

HADOOP VS SPARK

Hadoop	Spark
Map reduce is slow. Data is split, functions are mapped, and a reducer is needed. key/value pairs are needed to reconstruct everything.	Spark is 100x faster because it uses in-memory processing
Hadoop is meant for batch jobs	Spark can deal with continuous input/ouputs of data (data streams)
No pipelining to combine operations	Lazy evaluation (only when required) of multiple functions
Need to write lots of code (new functions from primitives and Java is verbose)	Scala is less verbose

https://intellipaat.com/blog/what-is-apache-spark/

APACHE SPARK ECOSYSTEM



https://medium.com/@rinu.gour123/apache-spark-ecosystem-complete-spark-components-guide-f3b57893173e

WHY HADOOP AND SPARK TOGETHER?

- Spark cannot completely replace Hadoop (and vice versa)
- Spark can run on top of Hadoop
- Spark MLlib has different ML functionality than Hadoop
- Spark does not have its own file system (it works well with HDFS)

SPARK MAIN CONCEPTS

- **RDD (Resilient Distributed Dataset).** RDDs are the main “data unit” of Spark. The data is distributed across cluster nodes and they can be used to perform distributed computing

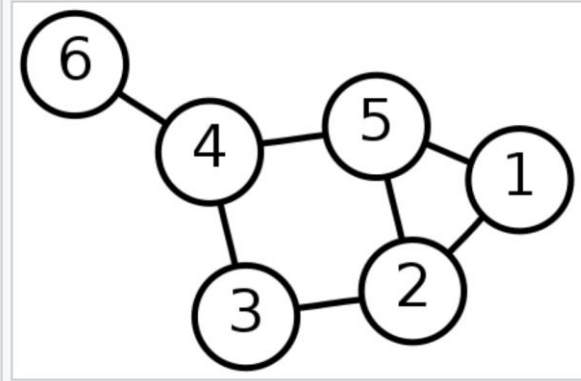
- **DAG (Direct Acyclic Graph).** DAGs are formed by vertices such that every edge is directed from initial to succeeding in the progression of operations.

https://www.kdnuggets.com/2018/10/apache-spark-introduction-beginners.html

DIRECT ACYCLIC GRAPHS

- Graph is a representation of a set of objects that have a relationship with one another

- The objects correspond to mathematical abstractions called vertices (or nodes or points) and each of the related pairs of vertices is called an edge (or link or line).

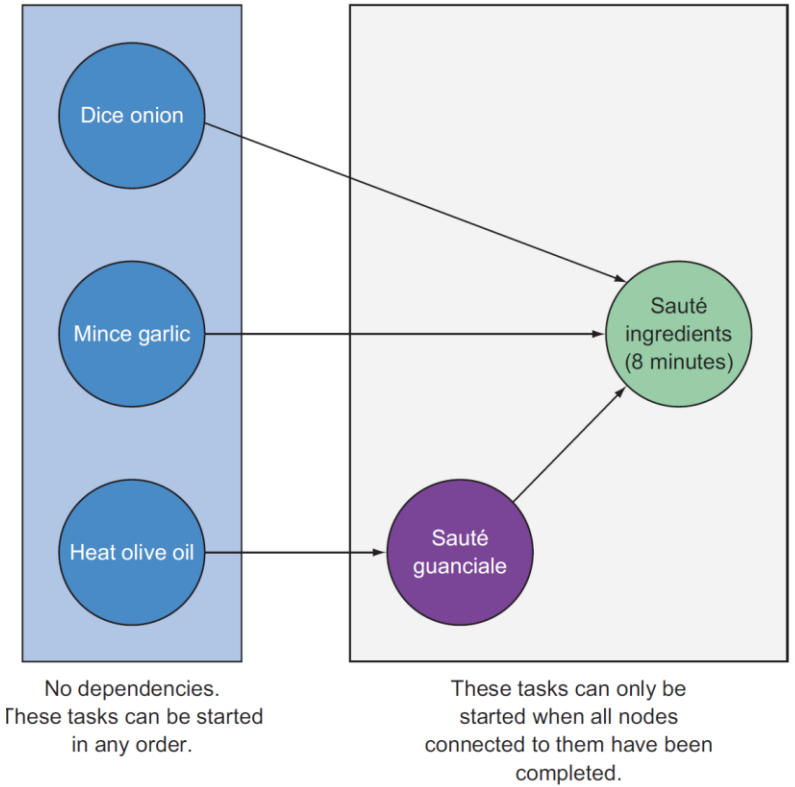


A graph with six vertices and seven edges.

Wikipedia

DIRECT ACYCLIC GRAPHS – COOKING

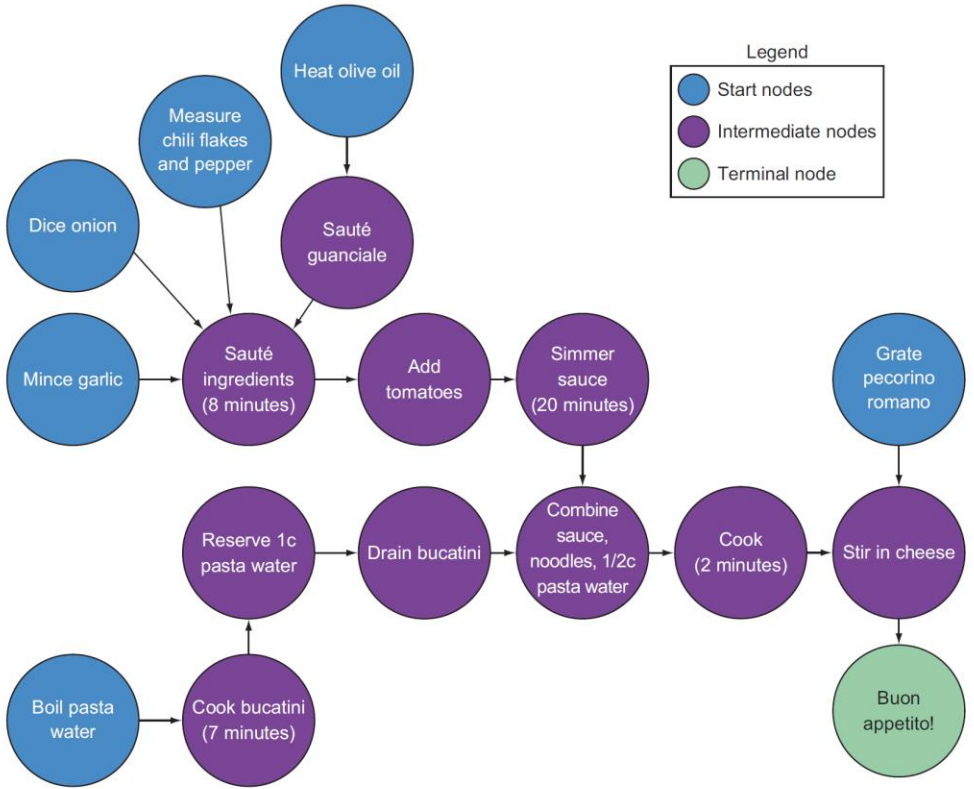
- Follow a series of sequential steps where raw ingredients are transformed into intermediate states until all the ingredients are ultimately combined into a complete dish



Jassie C. Daniel. Data Science with Python and Dask. Manning. 2019

DIRECT ACYCLIC GRAPHS – COOKING

- Full recipe view
- Scheduler / optimizing timing (e.g. put water to boil while dicing the onion).



Jassie C. Daniel. Data Science with Python and Dask. Manning. 2019

DAGS– *SCALING UP VS. SCALING OUT*

- *Scale up: 1 cook oversees the whole process. Better/faster equipment has a limit*
- *Scaling out: hire additional workers. Need new materials but cheaper than building specialized equipment*
- *DAGs are useful to distribute the work (and explain dependencies)*

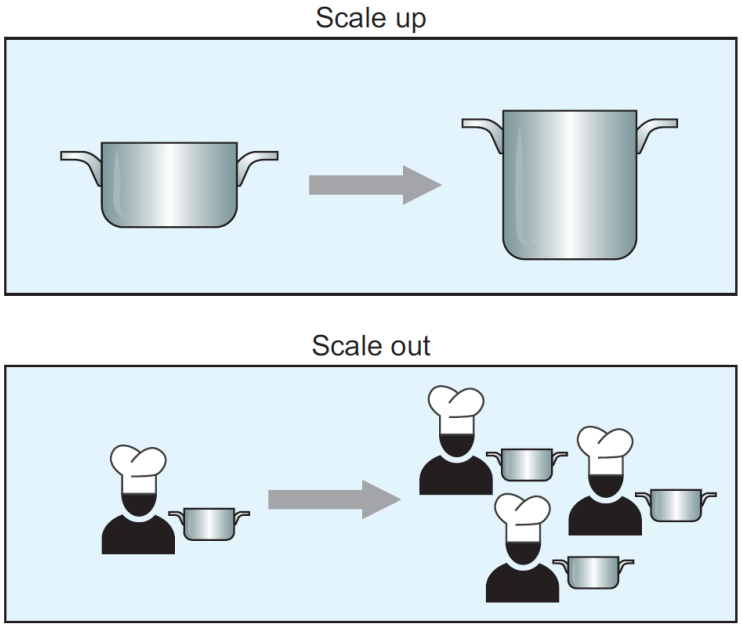


Figure 1.7 Scaling up replaces existing equipment with larger/faster/more efficient equipment, while scaling out divides the work between many workers in parallel.

DAGS– OPTIMIZING *SCALING OUT*

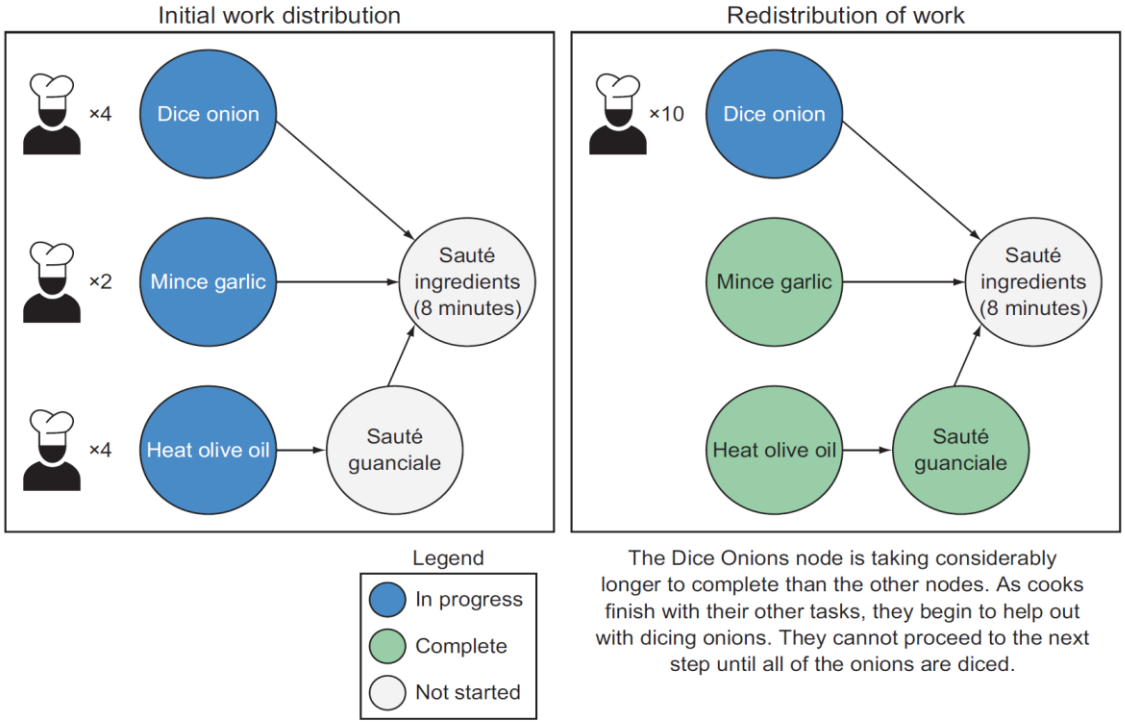


Figure 1.8 A graph with nodes distributed to many workers depicting dynamic redistribution of work as tasks complete at different times

DAGS– TASK SCHEDULER / CONCURRENCY

It is useless to hire more cooks than available knives.
If all knives are used for the onion. No one can cut the garlic

*The **task scheduler** considers **concurrency** and assigns resources (to avoid "fights" among cooks/computing nodes)*

DAGS– FAILURES

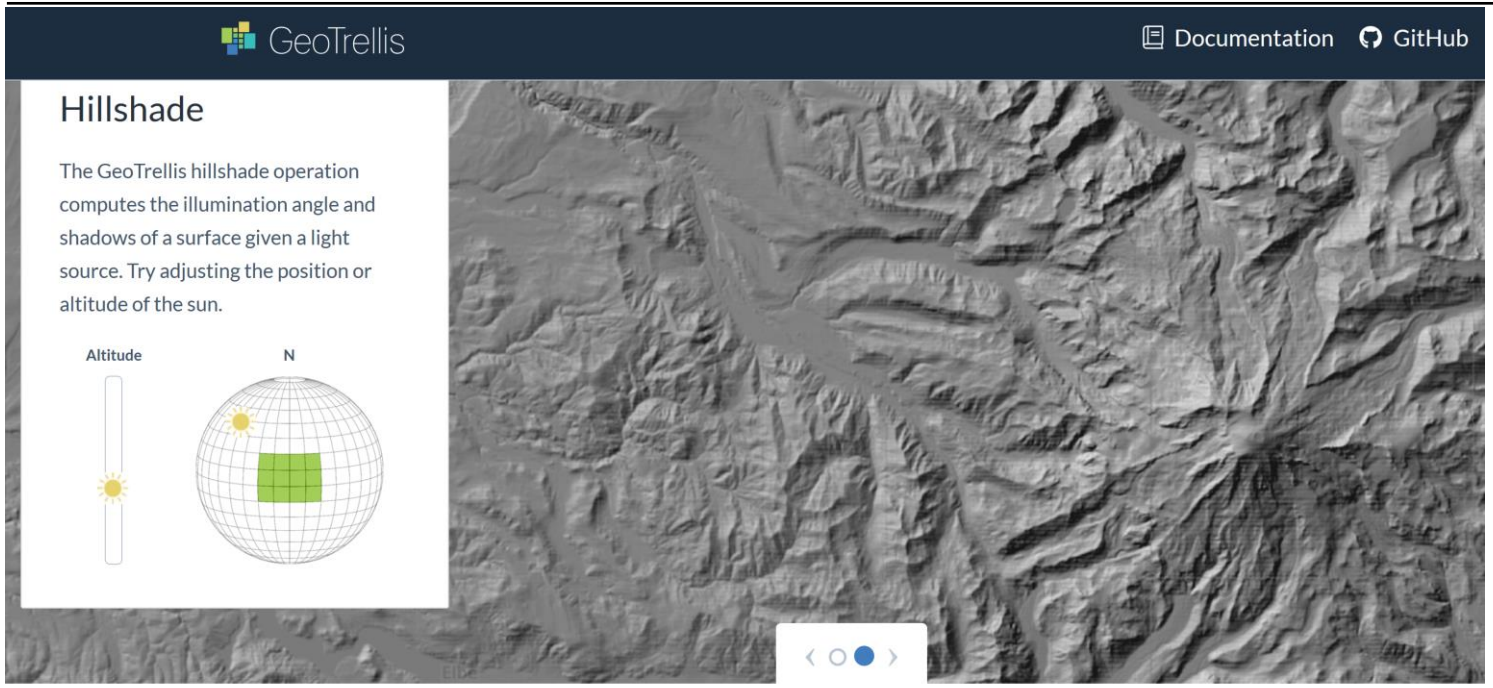
- Loose a worker: Cook quits but diced onion stays on the table… task scheduler sees this and can assign a new cook (if available) and tell him/her how much extra onion is needed.

- Loose data: boiling water pot falls. We need to start over again… time penalty. Go back to the DAG and find the first step without dependencies

The scheduler can fail too.. Then the whole DAG needs to be re-done.

SPARK-BASED SOLUTIONS

https://geotrellis.io/



GeoTrellis is a geographic data processing engine for high performance applications.

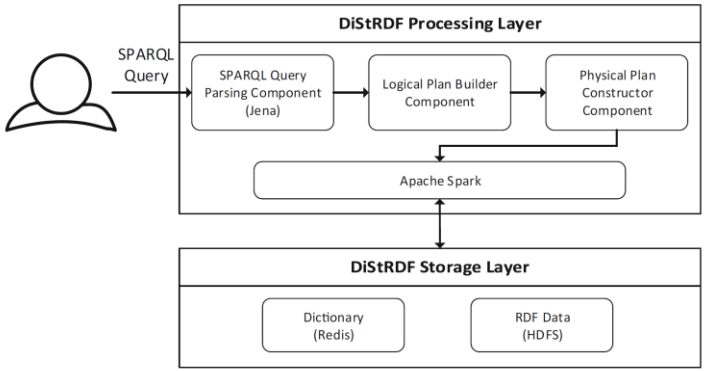
SPARK-BASED SOLUTIONS

Geoinformatica
https://doi.org/10.1007/s10707-019-00371-0

Parallel and scalable processing of spatio-temporal RDF queries using Spark

Panagiotis Nikitopoulos[1] · Akrivi Vlachou[1] · Christos Doulkeridis[1] · George A. Vouros[1]

Received: 2 August 2018 / Revised: 25 March 2019 / Accepted: 20 June 2019 /
Published online: 03 July 2019



SPARK-BASED SOLUTIONS

IEEE TRANSACTIONS ON BIG DATA, VOL. X, NO. X, JANUARY 2018

Exploring spring onset at c
mapping phenoregions
temperature and satellite-ba

Raul Zurita-Milla, Romulo Goncalves, Emma Izquierdo-

Abstract—Each spring many plants put on new leaves and/or open their flowers phenological data. Various phenological datasets can be used to study spring on novel exploratory analysis where we link two multi-decadal and high-spatial resol indices and land surface phenological metrics derived from satellite images. Our conterminous US, focuses on identifying regions with similar spring onset, and o products. Our results show that the spring onset patterns captured by the satellit temperature-based phenological indices. They also highlight areas with stable ar remain or change of phenoregion from year to year). Finally, our resutls reveal th negatively correlated with the phenological information that can be derived from to integrate multi-source phenological data. To cope with the computational chall our analysis on a cloud platform running Apache Spark and various of its extensi performed well and allowed the execution of user-tailored analyses. Hence, we b towards the efficient analysis of global vegetation phenology at very high spatial ever-increasing collections of geospatial data about our planet.

DASK

- Dask enables parallel and out-of-core computation in Python.
- Dask couples blocked algorithms with dynamic and memory aware task scheduling (parallel and out-of-core NumPy).

DASK

- Blocked algorithms provide tricks to work on a big data problem by solving very many small problems.
- "take the sum of these trillion numbers" with many small computations like "break up the trillion numbers into one million chunks of size one million, sum each chunk, then sum all of the intermediate sums."

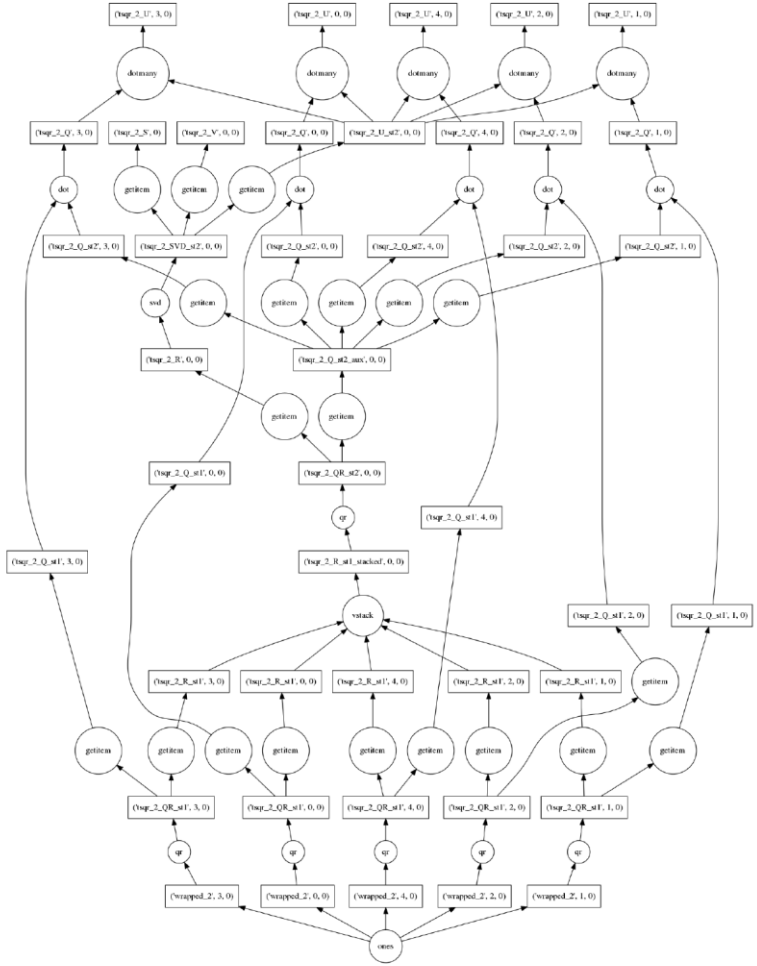


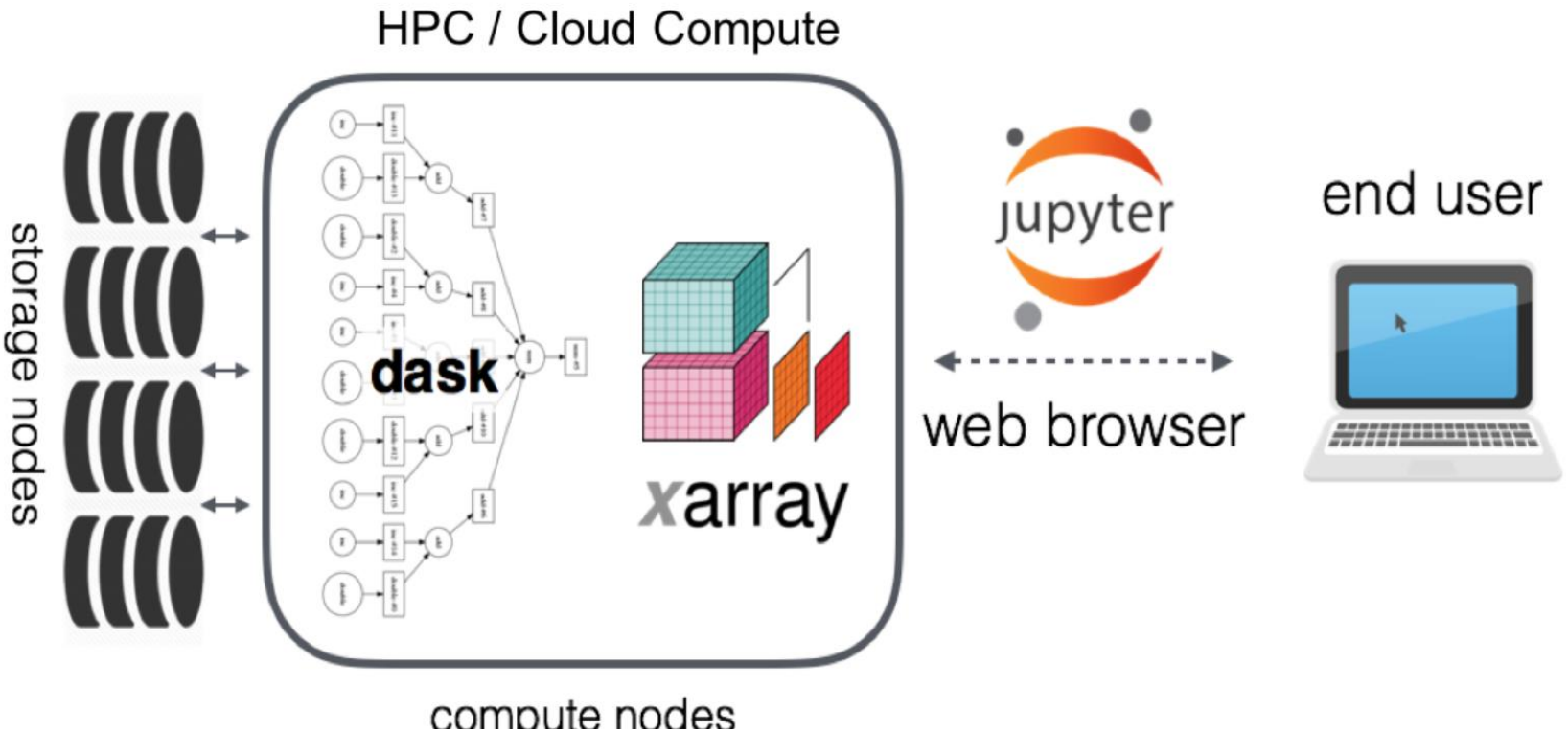
Fig. 4: Out-of-core parallel SVD

DASK DATA STRUCTURES

- A Dask Dataframe: formed by a set of small Pandas df. Most (but not all) Pandas functionality is available
- A Dask Bag: stores and process collections of Pythonic objects (log / JSON files)
- Dask Arrays: formed by a set of smaller Numpy arrays. Operations like slicing are allowed.

DASK-BASED SOLUTIONS

http://pangeo.io



DASK-BASED SOLUTIONS

PANGEO

INTERCHANGEABLE PIECES IN PANGEO (PICK 1 OR MORE FROM EACH ROW)

Data Models	xarray	Iris	pandas
N-D Arrays	NumPy	DASK	
Processing Mode	Interactive jupyter	Batch	Serverless
Compute Platform	HPC	aws	Google Cloud Platform
Foundation	python™		

ITC

DASK VS SPARK

DASK	SPARK
Written in Python. It interoperates well with C/C++/ Fortran/LLVM or other natively compiled code linked through Python	Written in Scala with some support for Python and R. It interoperates well with other JVM code.
A component of the larger Python ecosystem. It couples with and enhances other libraries: NumPy, Pandas, Scikit-Learn, …	An all-in-one project that has inspired its own ecosystem. It integrates well with many other Apache projects.

https://docs.dask.org/en/latest/spark.html

DASK VS SPARK

DASK	SPARK
Dask is younger and is an extension of the well trusted NumPy/Pandas /Scikit-learn/Jupyter stack.	Spark is older and has become a dominant and well-trusted tool in the Big Data enterprise world.
Applied more generally both to business intelligence applications, as well as several scientific and custom situations.	More focused on traditional business intelligence operations like SQL and lightweight machine learning.

DASK VS SPARK

DASK	SPARK
Able to implement sophisticated algorithms and build complex systems. It is fundamentally based on generic task scheduling.	Provides high-level optimizations on uniformly applied computations, but it lacks flexibility for complex algorithms/systems. It is fundamentally an extension of the Map-Shuffle-Reduce paradigm.
Scales from a single node to thousand-node clusters.	Scales from a single node to thousand-node clusters.

DASK VS SPARK

DASK	SPARK
Reuses the Pandas API and memory model. It implements neither SQL nor a query optimizer. It can do random access, efficient time series operations, and other Pandas-style indexed operations.	Has its own API and memory model. It also implements a large subset of the SQL language. Spark includes a high-level query optimizer for complex queries.
Relies on and interoperates with existing libraries like Scikit-Learn and XGBoost. These can be more familiar but it results in a less-cohesive solution.	MLLib is a cohesive project with support for common operations that are easy to implement with Spark's Map-Shuffle-Reduce style system.

DASK VS SPARK

DASK	SPARK
Fully supports the NumPy model for scalable multi-dimensional arrays.	Does not include support for multi-dimensional arrays natively. Some support for two-dimensional matrices can be found in MLLib. The Thunder project combines Spark with NumPy arrays.
Allows specifying arbitrary task graphs for more complex and custom systems that are not part of the standard set of collections.	Users are expected to compose computations out of high-level primitives (map, reduce, groupby, join, …).

WHY SPARK?

- You prefer Scala or the SQL language
- You have mostly a Java-based infrastructure
- You are mostly doing business analytics with some lightweight machine learning

WHY DASK?

- You prefer Python, or have large legacy code bases that you do not want to entirely rewrite
- Your use case is complex or does not cleanly fit the Spark computing model
- You want a lighter-weight transition from local computing to cluster computing
- You want to interoperate with other technologies and don’t mind installing multiple packages

WHY BOTH?

- It is easy to use Dask and Spark on the same data.
- Both read and write common formats, like CSV, JSON, making it easy to hand results off between Dask and Spark workflows.
- Most clusters can run many different distributed systems at the same time: If you already have a Spark cluster, it is easy to also run Dask workloads on your current infrastructure and vice versa.

OTHER BIG GEODATA SOLUTIONS

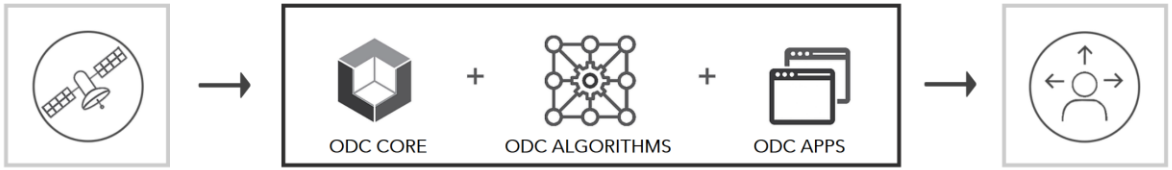
OPEN DATA CUBE

About Overview Install Applications Resources News Contact

The Open Data Cube (ODC) is an Open Source Geospatial Data Management and Analysis Software project that helps you harness the power of Satellite data. At its core, the ODC is a set of Python libraries and PostgreSQL database that helps you work with geospatial raster data. See our GitHub repository here>>.

The ODC seeks to increase the value and impact of global Earth observation satellite data by providing an open and freely accessible exploitation architecture. The ODC project seeks to foster a community to develop, sustain, and grow the technology and the breadth and depth of its applications for societal benefit.

ODC ECOSYSTEM
GEOSPATIAL DATA MANAGEMENT & ANALYSIS SOFTWARE



SATELLITE DATA
Examples:
- *Landsat*
- *Sentinel*
- *MODIS*

FLEXIBLE DEPLOYMENT
Depending on your application, the Open Data Cube can be deployed on HPC, Cloud, and local installations. Typical installations run on LInux, MacOS, and Windows.

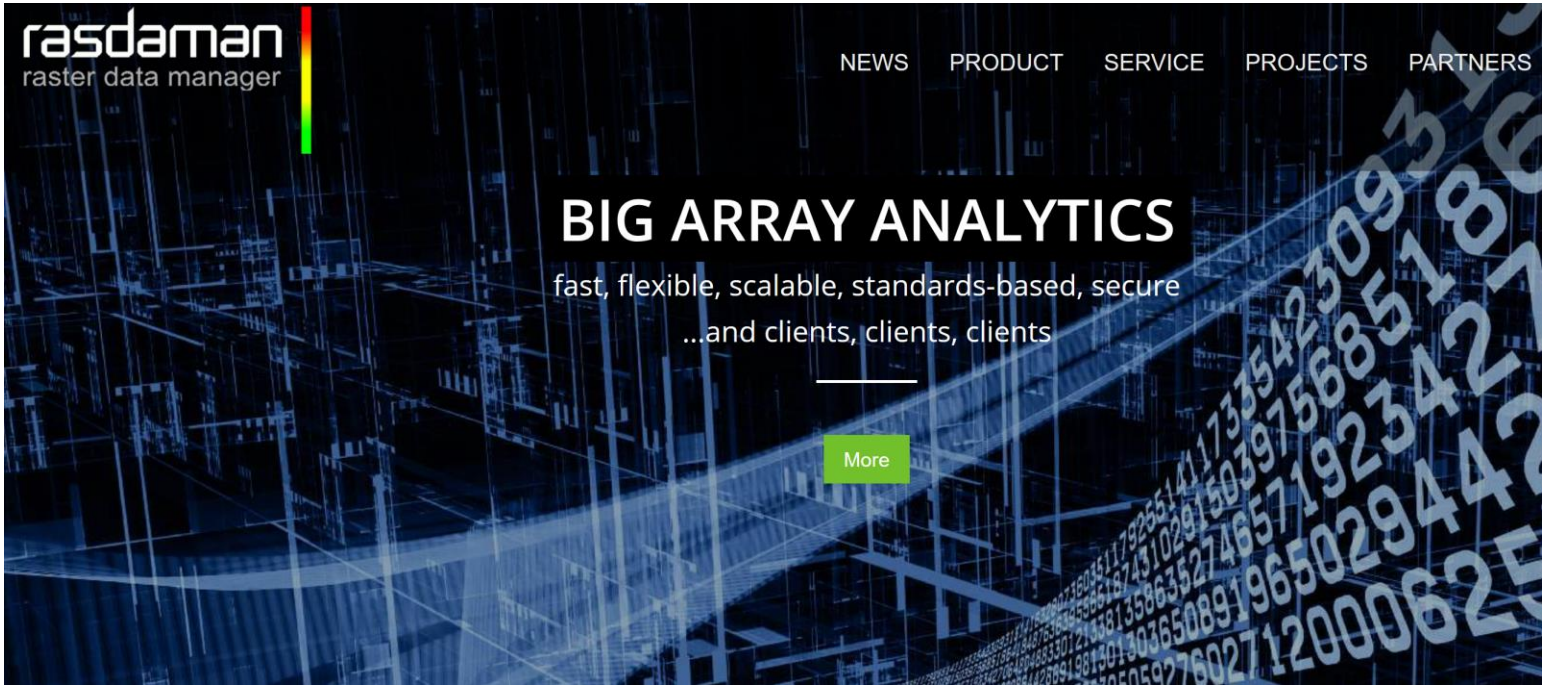
INFORMED DECISIONS
Examples:
- *Deforestation*
- *Water Quality*
- *Illegal Mining*

Learn More

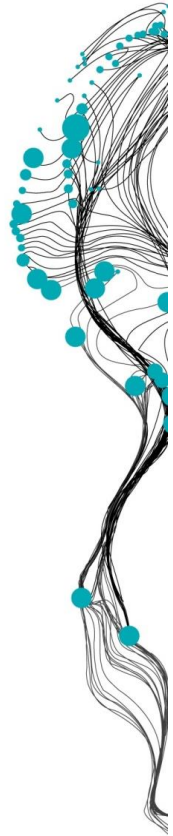
https://www.opendatacube.org/

ITC

OTHER BIG GEODATA SOLUTIONS (II)



https://www.rasdaman.com/



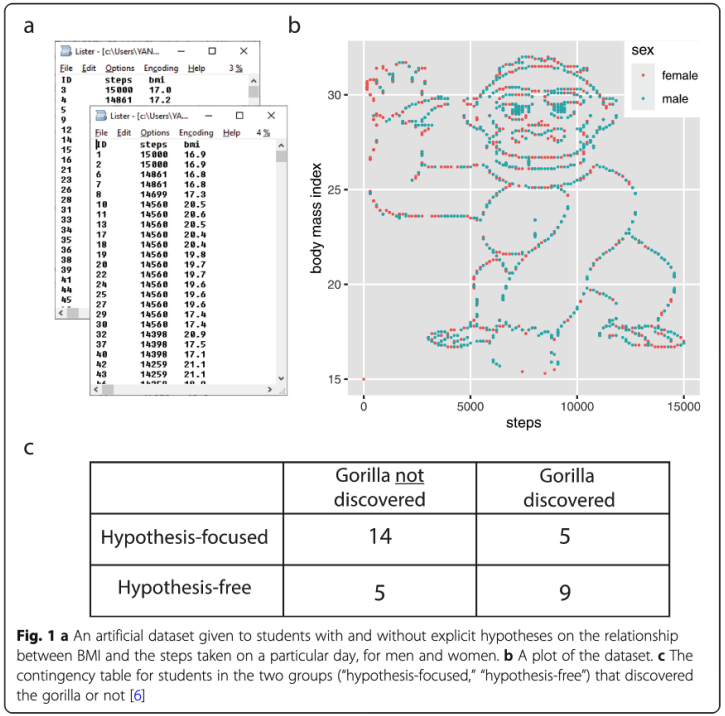
DO YOU KNOW OF OTHER BIG (GEO)DATA SOLUTION?

PRACTICALS

“ ‘When someone seeks,’ said Siddhartha, ‘then it easily happens that his eyes see only the thing that he seeks, and he is able to find nothing, to take in nothing. [...] Seeking means: having a goal. But finding means: being free, being open, having no goal.’ ” Hermann Hesse

- Exploring your data using regular Python and Dask

One thing we have learned from decades of exploratory data analysis: do not give up on a dataset. If it does not support your original hypothesis, it likely contains hints at alternative, possibly even more interesting phenomena. And if the data supports your original hypothesis, still keep exploring beyond. If the dataset has been designed and assembled well, there are likely additional discoveries to be made. These cannot be expected to emerge after just a first look. They will take time to unfold. It is not well appreciated, but the truth is that one never really finishes to analyze a dataset. You just decide to stop and move on at some point, leaving some things undiscovered….



c

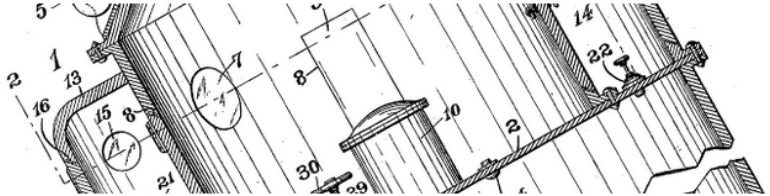
	Gorilla not discovered	Gorilla discovered
Hypothesis-focused	14	5
Hypothesis-free	5	9

Fig. 1 a An artificial dataset given to students with and without explicit hypotheses on the relationship between BMI and the steps taken on a particular day, for men and women. **b** A plot of the dataset. **c** The contingency table for students in the two groups (“hypothesis-focused,” “hypothesis-free”) that discovered the gorilla or not [6]

https://doi.org/10.1186/s13059-020-02133-w

PRACTICALS

- Contrasting views…



Patent #1,059,281 (Diving Apparatus for Marine Exploration and the Like)

Banning exploration in my infovis class

Eytan Adar Apr 26, 2017 · 9 min read

I've banned the word "explore" from all project proposals in my infovis class. No *explore*. No *exploration*. No *exploratory*. No, you may not create a tool to "allow an analyst to explore the bird strike data." No, you can't build a system for "exploration of microarray data." And, no, you can't make a framework for "exploratory network analysis." Just no.

The line that I use on my students is that: *No one is paid to explore, they're paid to find*. I'm only 10% trying to be clever. Ninety percent, I'm dreading grading the output of projects that feature exploration as an objective.

https://medium.com/@eytanadar/banning-exploration-in-my-infovis-class-9578676a4705