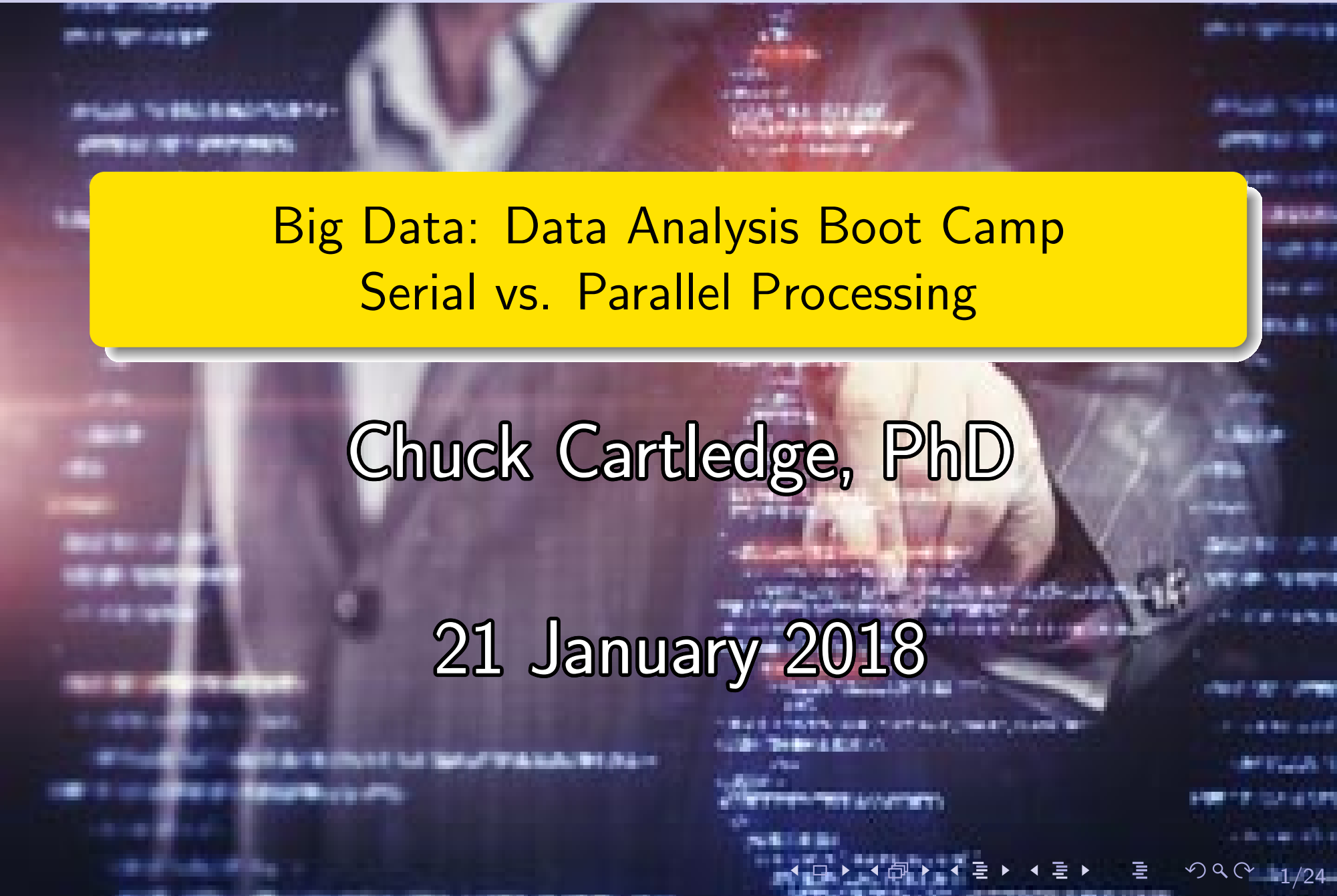


Big Data: Data Analysis Boot Camp
Serial vs. Parallel Processing

Chuck Cartledge, PhD

21 January 2018

Table of contents (1 of 1)

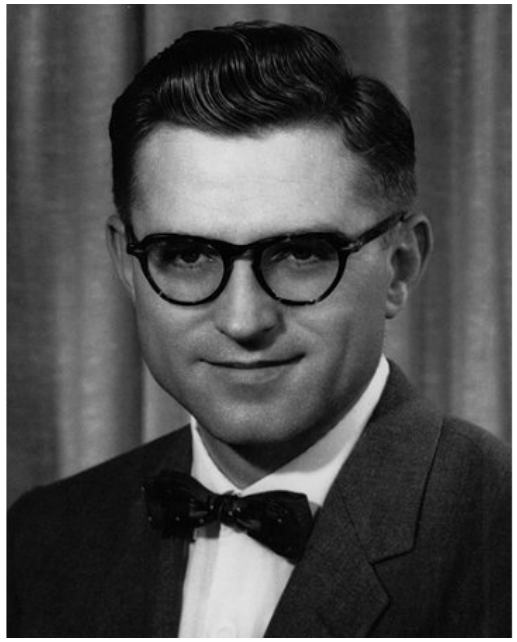
What are we going to cover?

We're going to talk about:

- Why it is important to understand your problem
- What are single and multithreaded programs
- What are different tools, and frameworks to support BD processing
- What languages and programming paradigms fit the BD world

Amdahl's Law [2]

- Time for serial execution $\stackrel{\text{def.}}{=} T(1)$
- Portion that can NOT be paralyzed $\stackrel{\text{def.}}{=} B \in [0, 1]$
- Number of parallel resources $\stackrel{\text{def.}}{=} n$
- $T(n) = T(1) * (B + \frac{1}{n}(1 - B))$
- Speed up $\stackrel{\text{def.}}{=} S(n)$
  $S(n) = \frac{T(1)}{T(n)} = \frac{1}{B + \frac{1}{n}(1 - B)}$

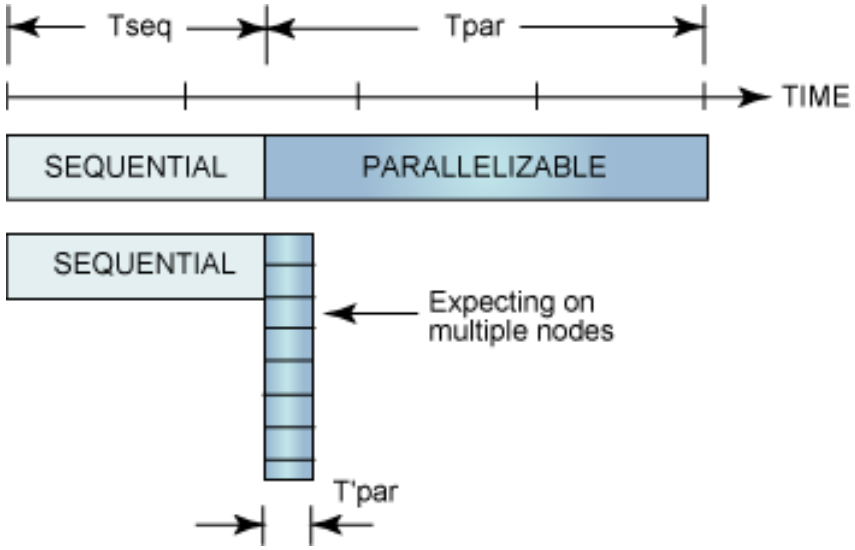


Dr. Gene Amdahl (circa 1960)

Amdahl's Law (A summary)

Division and measurement of serial and parallel operations appears time and again. (Shades of Mandelbrot.)

- "Make the common fast."
- "Make the fast common."
- Understand what parts have to be done serially
- Understand what parts can be done in parallel

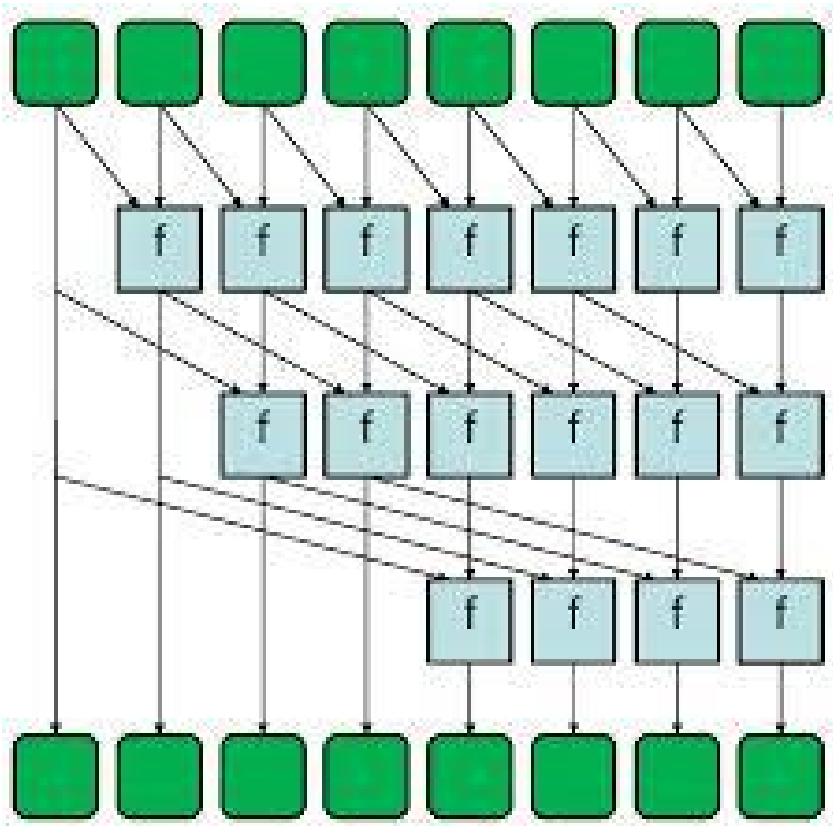


Need to factor in "overhead" costs when computing speed up.

Some questions are easily stated, . . .

Which of these are paralizable (and why)?

1. $a[i] = b[i] + c[i]$
2. $a[i] = f(b)$
3. $a[i] = a[i-1] + b[i-1]$
4. $a = b + c$



Single thread vs. multithreads

- Single-threaded process – has full access to CPU and RAM
- Multithreaded process – shares access to CPU and RAM
- Multithreaded makes sense with independent tasks
- Multithreaded may share the same memory space (language dependent)

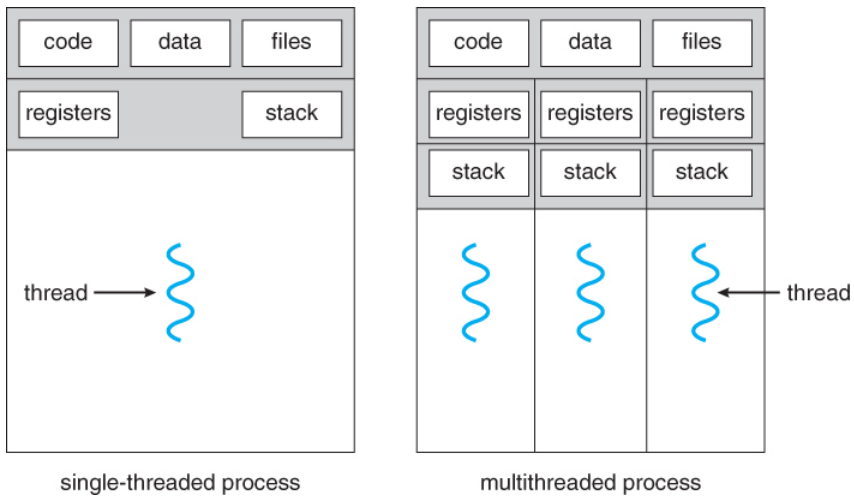


Image from [3].

Coordination across multiple threads can be tricky.

Hadoop multithreading hidden from view.

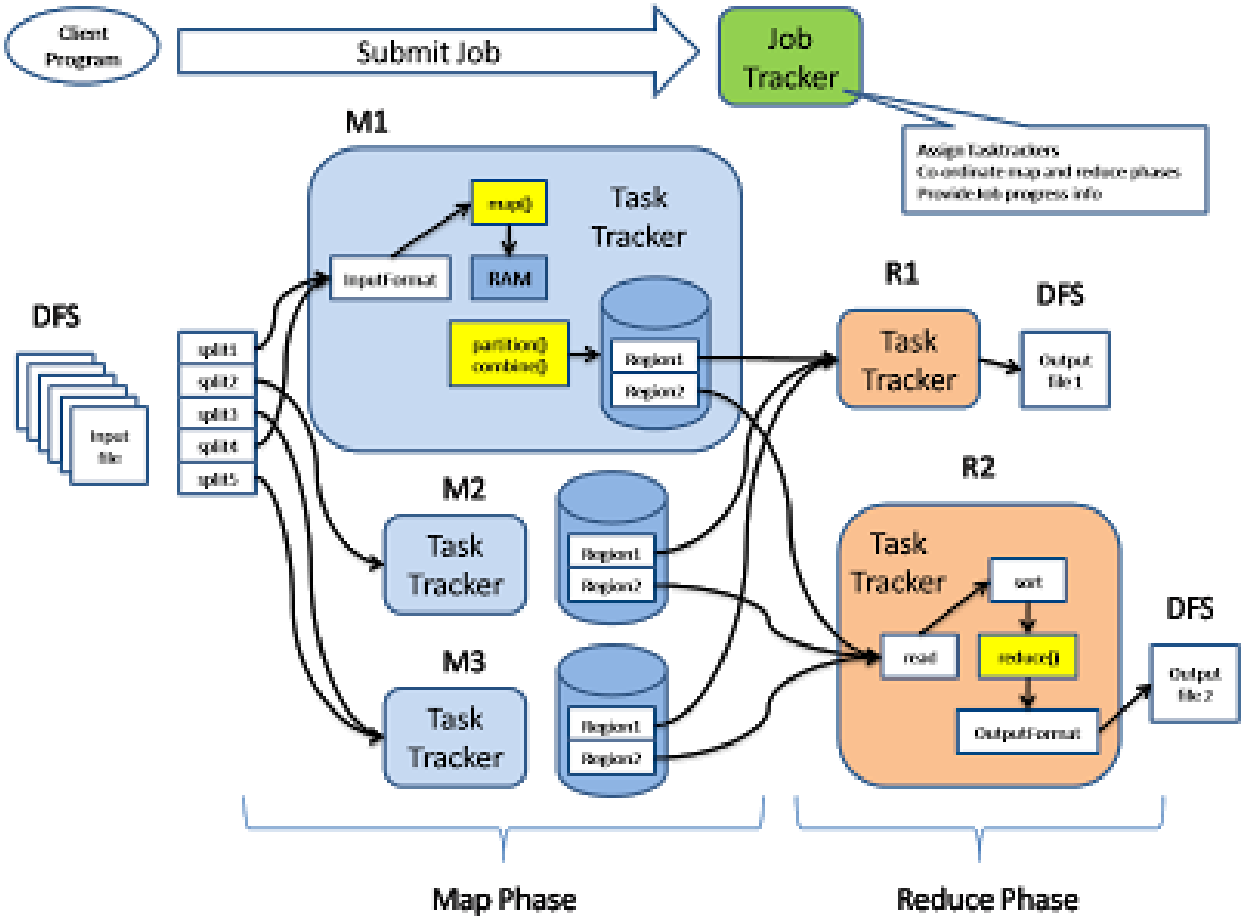


Image from [4].

Vocabulary

- Data Sources – where data comes from
- Ingestion – how data is pre-processed for acceptance
- Data Sea/Lake – where data lives
- Processing – how data is processed prior to storage
- Data warehouse – transition from SQL to NoSQL
- Analysis – extracting information from data
- User interface – how the user interacts with the information

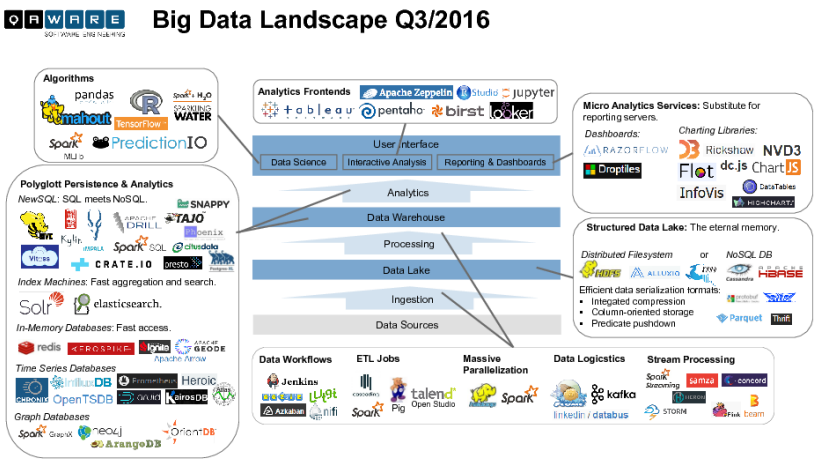


Image from [1].

Same image.

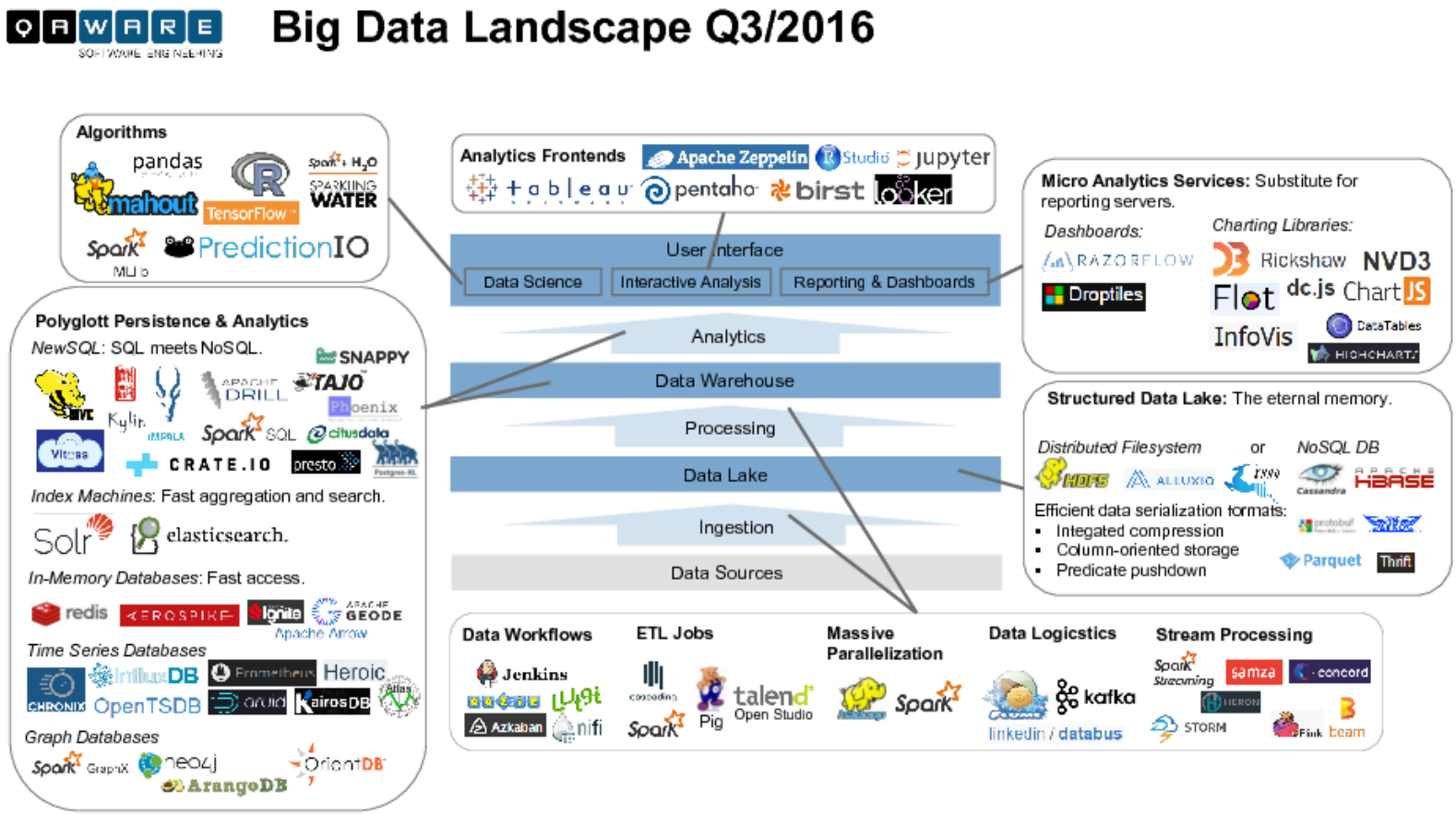


Image from [1].

Another collection of Open Source BD tools

Tools partitioned differently:

- Big Data search
- Business Intelligence
- Data aggregation
- Data Analysis & Platforms
- Databases / Data warehousing
- Data aggregation
- Data Mining
- Document Store
- Graph databases
- Grid Solutions
- In-Memory Computing
- KeyValue
- Multidimensional
- Multimodel
- Multivalue database
- Object databases
- Operational
- Programming
- Social
- XML Databases

Image from [7].

Image from [7].

The reference has links to each piece of software.

Hammer and nails . . .

". . . it is tempting, if the only tool you have is a hammer, to treat everything as if it were a nail."

Abraham H. Maslow [6]

Image from [8].

A simple comparison of some languages

Languages are created by people to solve certain types of problems.

- C# – declarative, functional, generic, imperative, object-oriented (class-based)
- Java – client side, compiled, concurrent, curly-bracket, impure, imperative, object-oriented (class-based), procedural, reflective
- Python – compiled, extension, functional, imperative, impure, interactive mode, interpreted, iterative, metaprogramming, object-oriented (class-based), reflective, scripting
- R – array, impure, interpreted, interactive mode, list-based, object-oriented prototype-based, scripting

Categorizations from [9].

Vocabulary (1 of 2)[9].

- array – generalize operations on scalars to apply transparently to vectors, matrices, and higher-dimensional arrays.
- client side – languages are limited by the abilities of the browser or intended client.
- compiled – languages typically processed by compilers, though theoretically any language can be compiled or interpreted.
- concurrent – languages provide language constructs for concurrency.
- curly-bracket – languages have a syntax that defines statement blocks using the curly bracket or brace characters
- declarative – languages describe a problem rather than defining a solution.
- extension – languages embedded into another program and used to harness its features in extension scripts.
- functional – languages define programs and subroutines as mathematical functions.
- generic – language is applicable to many domains.
- imperative – languages may be multi-paradigm and appear in other classifications.
- impure – languages containing imperative features.
- interactive mode – languages act as a kind of shell

Vocabulary (2 of 2)[9].

- interpreted – languages are programming languages in which programs may be executed from source code form, by an interpreter.
- iterative – languages are built around or offering generators.
- list-based – languages are a type of data-structured language that are based upon the list data structure.
- metaprogramming – that write or manipulate other programs (or themselves) as their data or that do part of the work that is otherwise done at run time during compile time.
- object-oriented (class-based) – support objects defined by their class.
- object-oriented prototype-based – languages are object-oriented languages where the distinction between classes and instances has been removed
- procedural – languages are based on the concept of the unit and scope
- reflective – languages let programs examine and possibly modify their high level structure at runtime.
- scripting – another term for interpreted

Each reflects/supports a programming paradigm

A plethora of programming paradigms:

- Action
- Agent-oriented
- Automata-based
- Concurrent
- Data-driven
- Declarative
- Functional
- Dynamic
- Event-driven
- Generic
- Imperative
- Language-oriented
- Parallel
- Semantic
- Structured

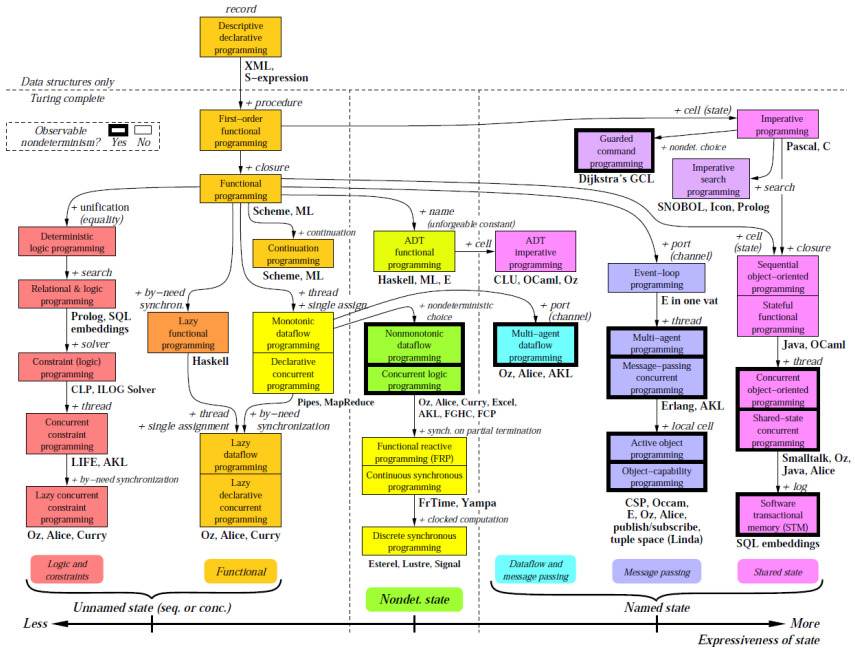


Image from [10].

Each paradigm is a result of a problem domain.

Each is different for a reason

Same image.

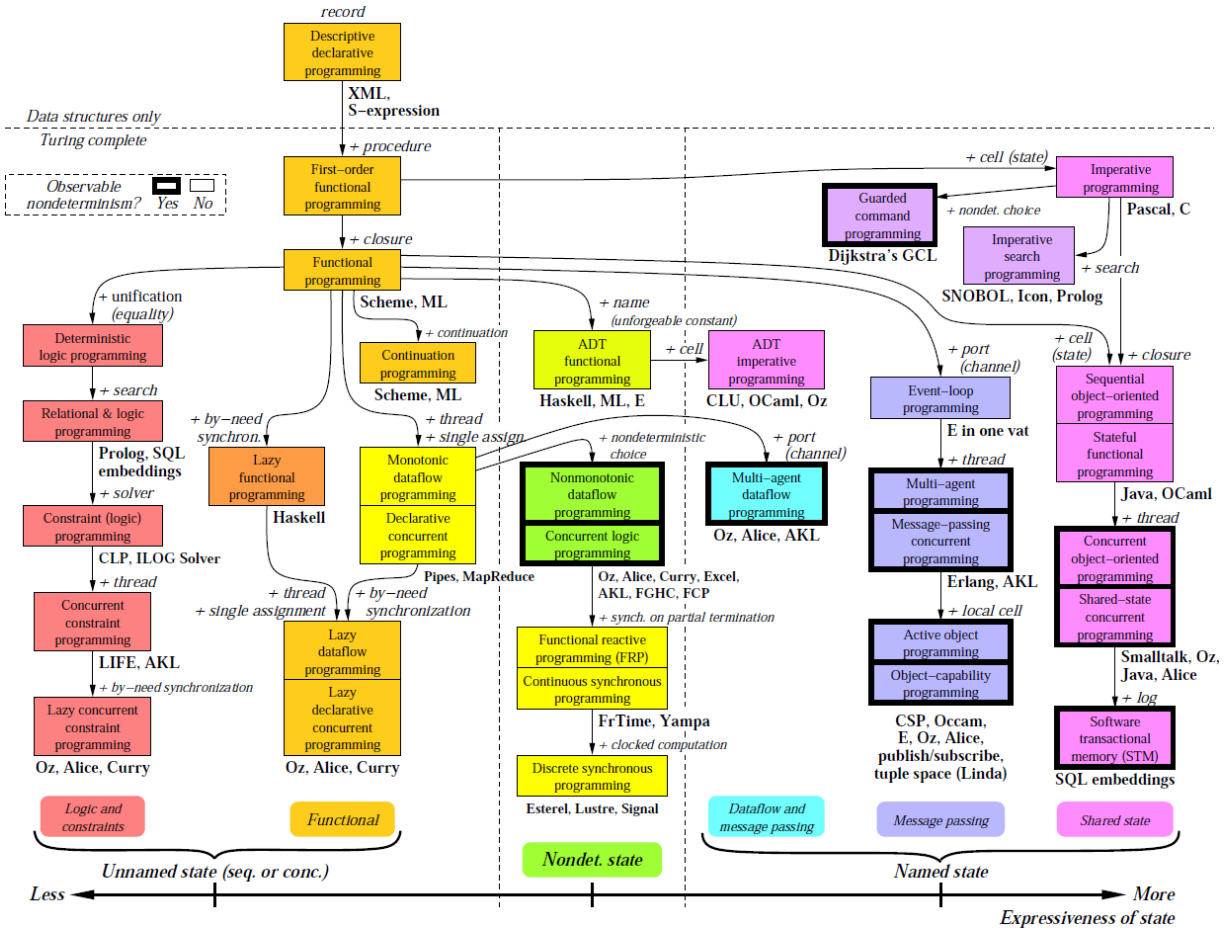


Image from [10].

What does the future hold?

> *"If languages are not defined by taxonomies, how are they constructed? They are aggregations of features. Rather than study extant languages as a whole, which conflates the essential with the accidental, it is more instructive to decompose them into constituent features, which in turn can be studied individually. The student then has a toolkit of features that they can re-compose per their needs."*
>
> *S. Krishnamurthi [5]*

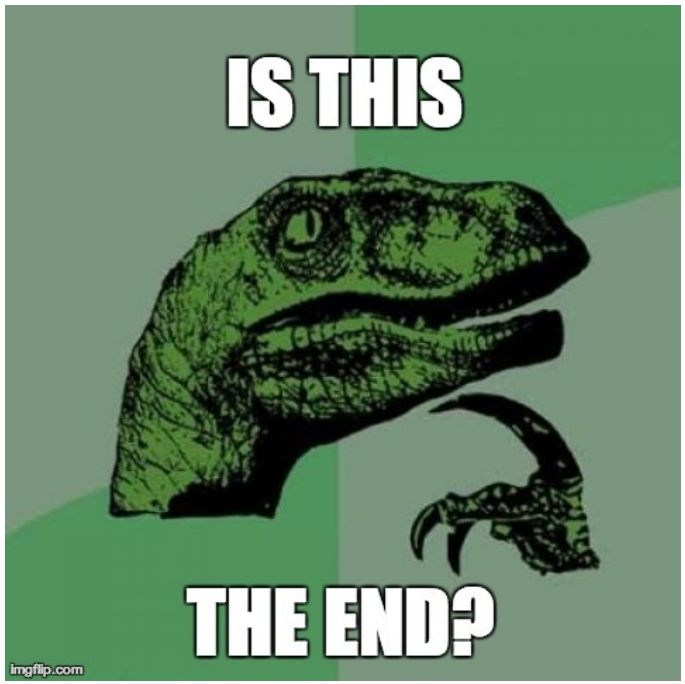
New languages will be created all the time to fit needs.

Q & A time.

"Never attempt to teach a pig to sing; it wastes your time and annoys the pig."
Robert A. Heinlein, Time Enough for Love

What have we covered?

- The importance of understanding your problem
- The importance of knowing the limitations of your tools
- The importance of understanding marketing hype



Next: BDAR Chapter 3, unleashing R

References (1 of 3)

[1] Josef Adersberger, Big Data Landscape Q3/2016, email, 2016.

[2] Gene M Amdahl, Validity of the single processor approach to achieving large scale com Proceedings of the Spring Joint Computer Conference, ACM, 1967, pp. 483–485.

[3] John T. Bell, Threads, `https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/4_Threads.html`, 2013.

[4] Ricky Ho, How Hadoop Map/Reduce works, 2008.

[5] Shriram Krishnamurthi, Teaching Programming Languages in a Post-Linnaean Age, SIGPLAN Notices **43** (2008), no. 11, 81–83.

References (2 of 3)

[6] Abraham H. Maslow, The Psychology of Science, Henry Regency, 1966.

[7] DataFloq Staff, The Big Data Open Source Tools Landscape, https://datafloq.com/big-data-open-source-tools/os-home/, 2014.

[8] Happiness Staff, Abraham Maslow, http://www.pursuit-of-happiness.org/history-of-happiness/abraham-maslow/, 2016.

[9] Wikipedia Staff, List of programming languages by type, https://en.wikipedia.org/wiki/List_of_programming_languages_by_type, 2106.

References (3 of 3)

[10] Peter Van Roy et al.,
Programming Paradigms for Dummies: What Every Programmer Sh
New Computational Paradigms for Computer Music **104**
(2009).