Scalable, Interactive, and Reproducible Data Mining of 3D Macromolecular Structures

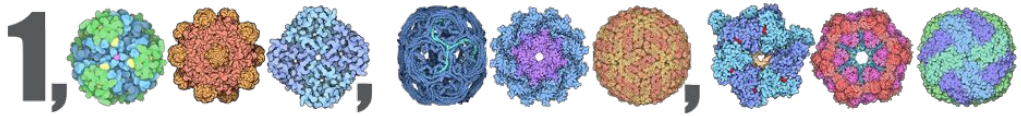
Peter Rose (pwrose@ucsd.edu)
Director, Structural Bioinformatics Lab
San Diego Supercomputer Center
UC San Diego

Outline

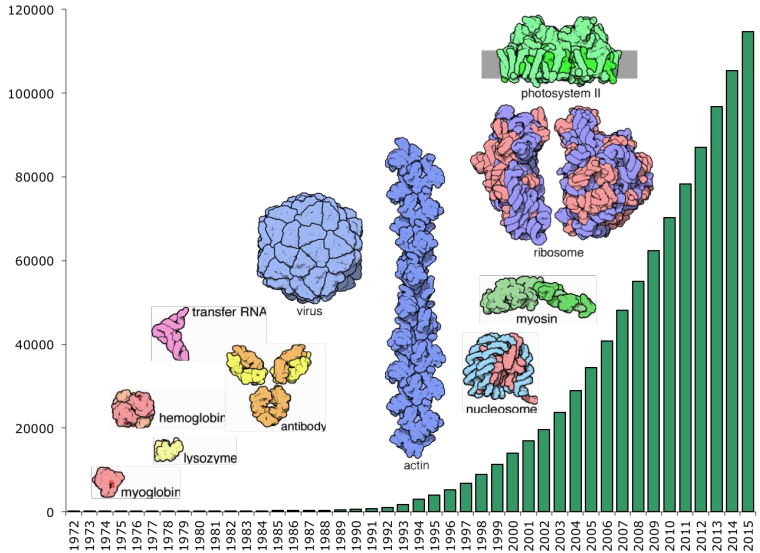
- Introduction
- Compressive Structural Bioinformatics
- Scaling up using Spark
- Interactive Exploration in Jupyter Notebook
- Working with Dataframes
- Deploying SPARK
- Application: map SARS-CoV-2 mutations

Introduction

PDB – A Billion Atom Archive



~1.4 billion atoms in the asymmetric units

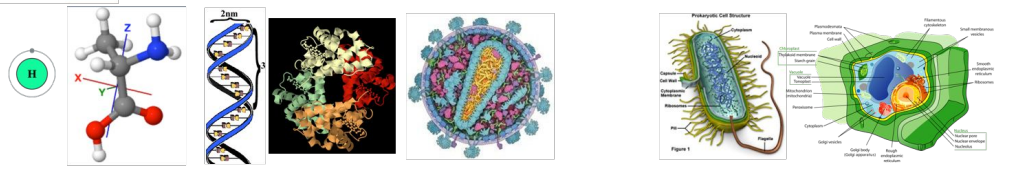


~177,000 structures in April 2021

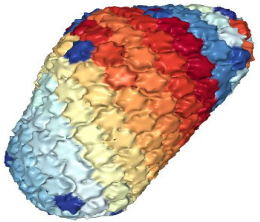
~1.8 million files downloaded per day!

> 1 million users per year

Growing Structure Size and Complexity

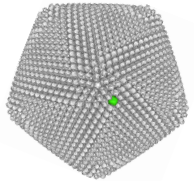


Largest asymmetric structure in PDB



HIV-1 capsid: PDB ID 3J3Q
~2.4M unique atoms

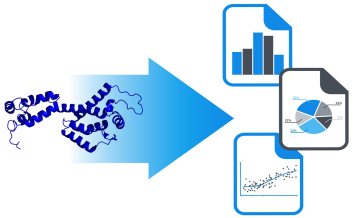
Largest symmetric structure in PDB



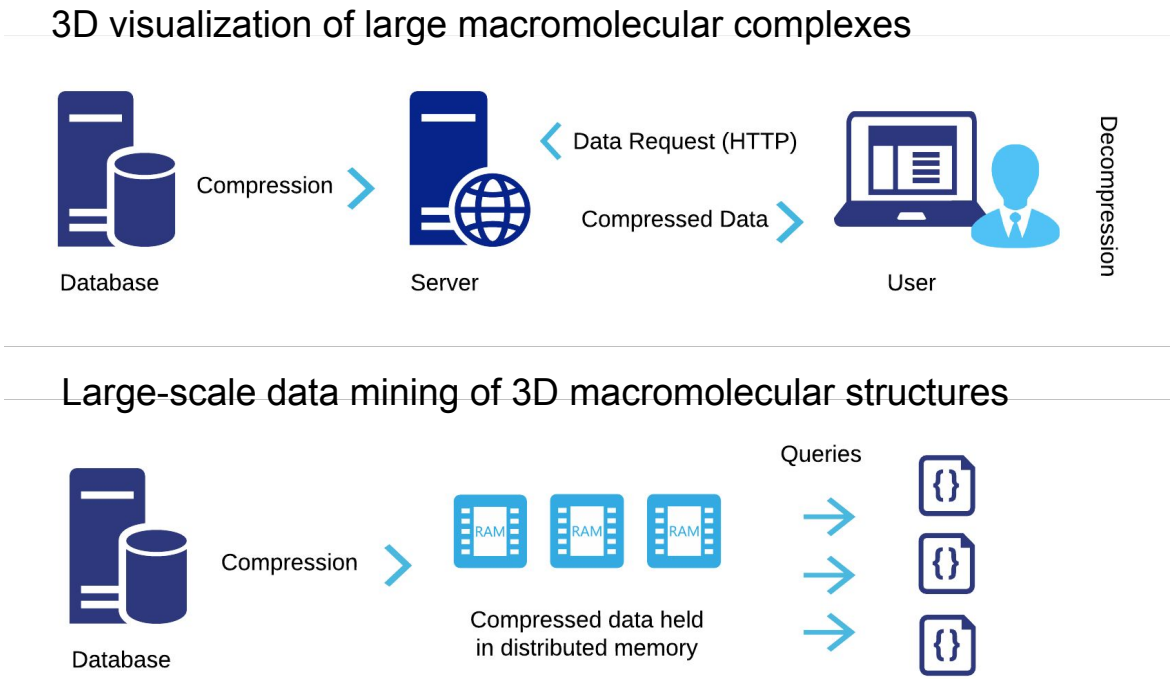
Faustovirus major capsid: PDB ID 5J7V
~40M overall atoms

Scalability Issues

- **Interactive visualization**
  - slow network transfer
  - slow parsing
  - slow rendering
- **Mobile visualization**
  - limited bandwidth
  - limited memory
- **Large-scale structural analysis**
  - slow repeated I/O
  - slow repeated parsing

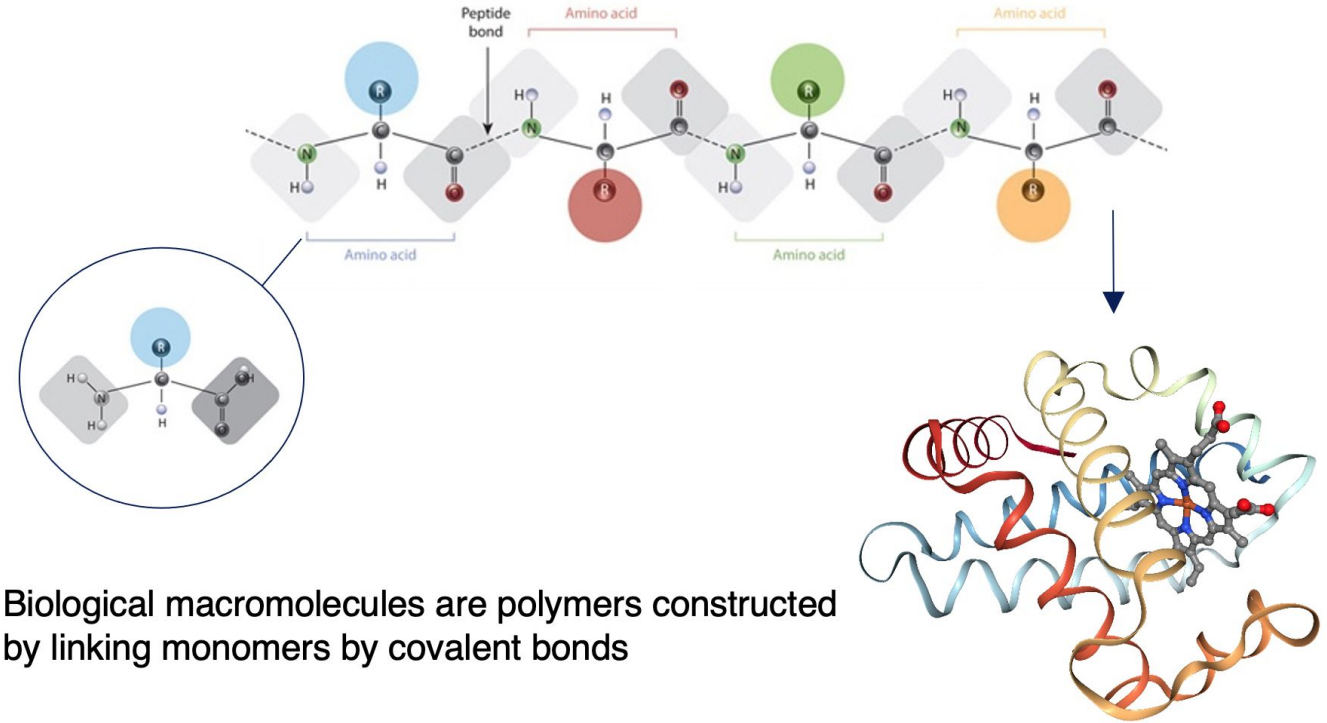


Compressive Structural Bioinformatics (MMTF Macromolecular Transmission Format)



Macromolecular 3D Structures

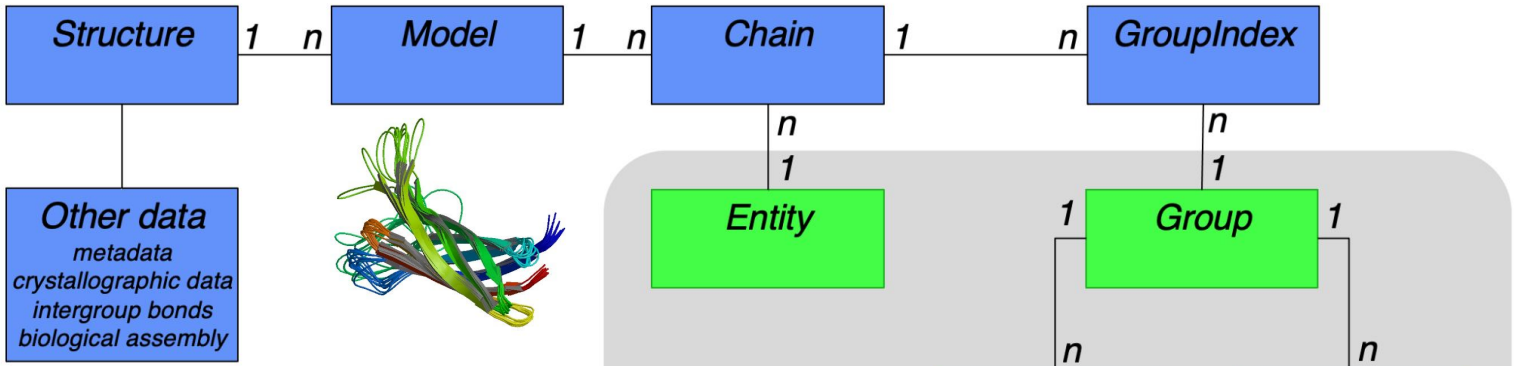
Biological macromolecules: proteins, nucleic acids



Biological macromolecules are polymers constructed by linking monomers by covalent bonds

MMTF Datastructure

Flat (columnar encoded) data structure with an implicit hierarchy

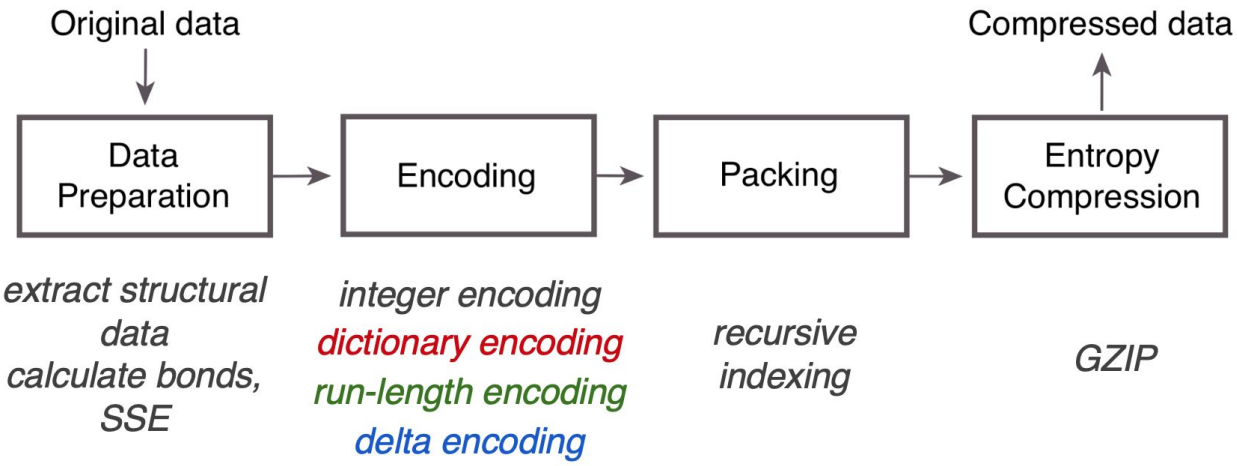


Entity
A unique molecule in a structure,
e.g., polymer chain, ligand, or water

Group
A unique chemical group (residue)

unique entities and groups are stored only once
e.g., 20 natural amino acids, water

MMTF Compression Pipeline

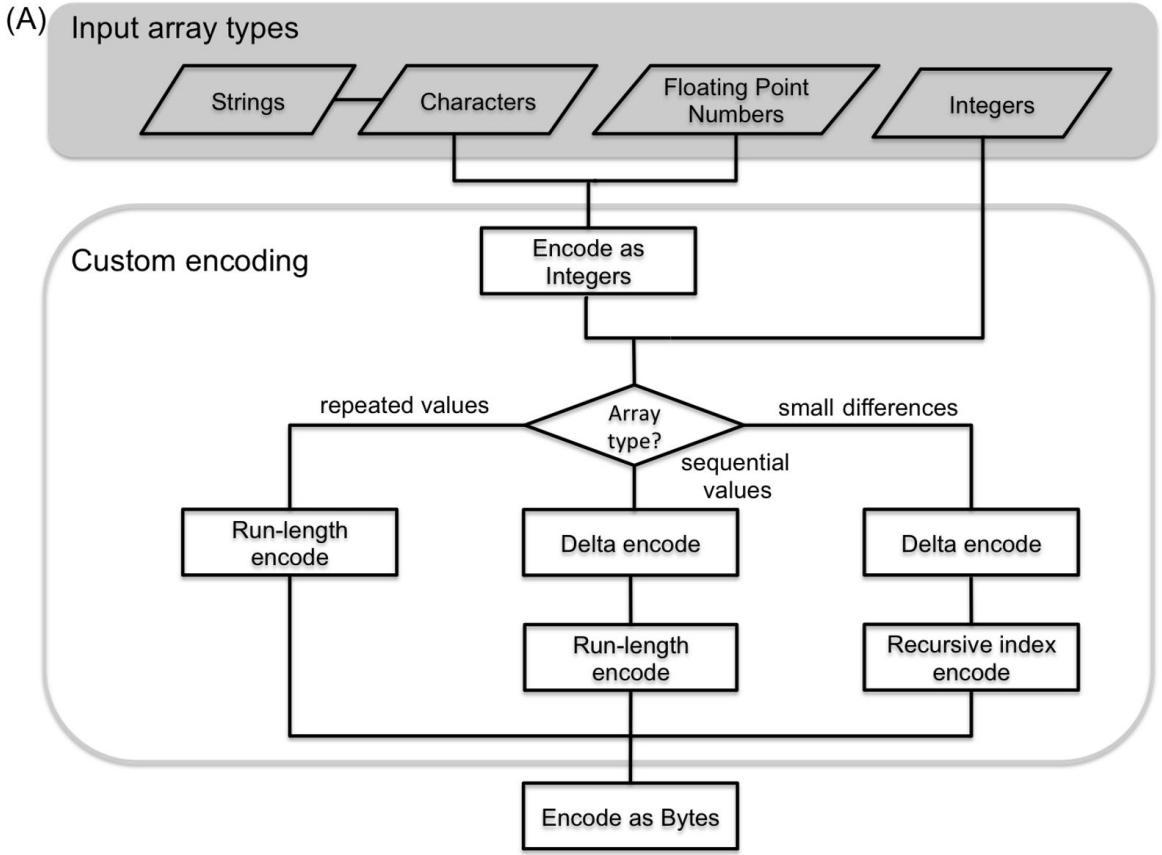


Binary, extensible container format of MMTF

MessagePack

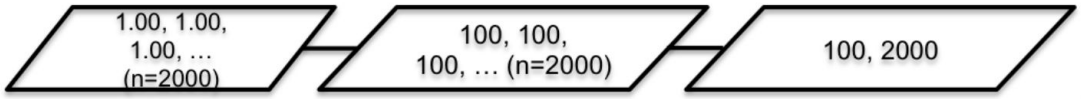
It's like JSON.
but fast and small.

Columnar Encoding of Arrays

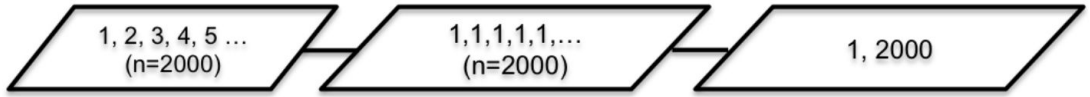


Encoding Examples

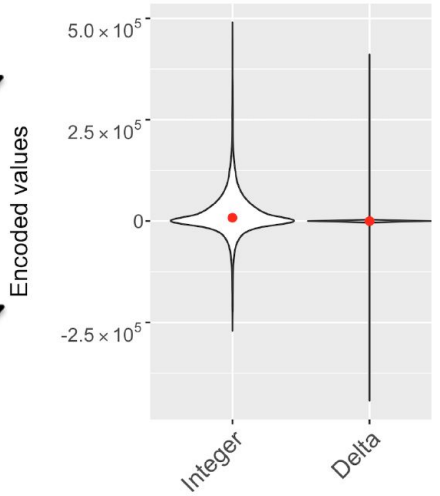
(B) Repeated values — *Run-length encoding (e.g., occupancy)*



(C) Sequential values — *Delta and run-length encoding (e.g., serial numbers)*



(D) Small differences — *Integer, delta, and recursive index encoding (e.g., xyz-coordinates as 16-bit integers)*



Delta encoding reduced the dynamic range of numbers which makes them more compressible

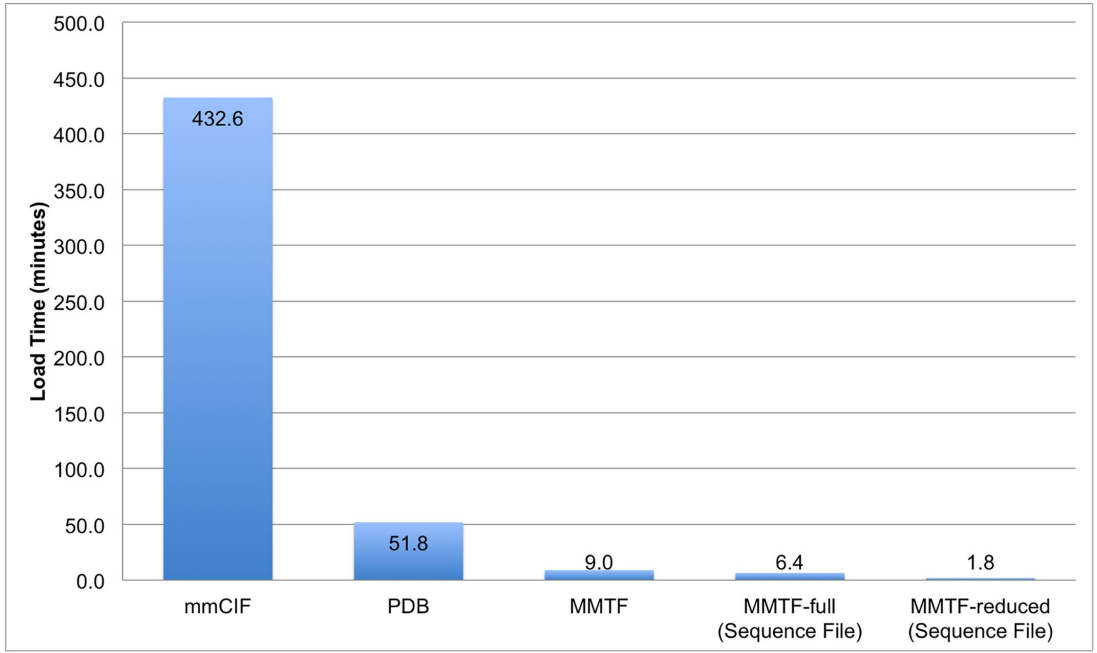
Dictionary Encoding

Unique groups (residues) are stored in a "dictionary"


```
{
        "groupName": "SER",
        "singleLetterCode": "S",
        "chemCompType": "L-PEPTIDE LINKING",
        "atomNameList": [ "N", "CA", "C", "O", "CB", "OG" ],
        "elementList": [ "N", "C", "C", "O", "C", "O" ],
        "formalChargeList": [ 0, 0, 0, 0, 0, 0 ],
        "bondAtomList": [ 1, 0, 2, 1, 3, 2, 4, 1, 5, 4 ],
        "bondOrderList": [ 1, 1, 2, 1, 1 ]
}
```


Unique entities are stored in a "dictionary"
(e.g., polymer type, polymer sequences)

File Parsing Speed (PDB archive ~127,000 entries)



Comparison made with BioJava and mmtf-java API (single core)

Download and Parsing Speed

Time (seconds) to download* 100 large PDB structures from UCSD and parse with JavaScript decoder in Chrome browser

*Note: download times are highly variable and not representative

Demo:
Download
Visualization
Video
https://mmtf.rcsb.org

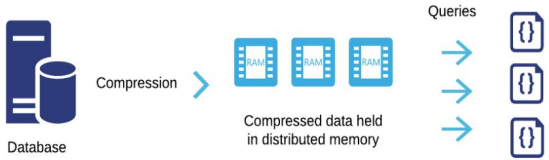
Summary

- **MacroMolecular Transmission Format (MMTF, mmtf.rcsb.org)**
  - Compressed, binary, efficient representation of 3D structures
    - Lossless representation (~4x compression over gzip)
    - Lossy, reduced representation (~37x compression over gzip)

- **Compressive Structural Bioinformatics**
  - Algorithms, application, and workflows using MMTF
    - 10 to 100+ fold speedup

Structure Visualization

Large Scale PDB Mining



Web-based molecular graphics for large complexes (2016)
Web 3D '16, 185-186, DOI: 10.1145/2945292.2945324

Enable Scaling

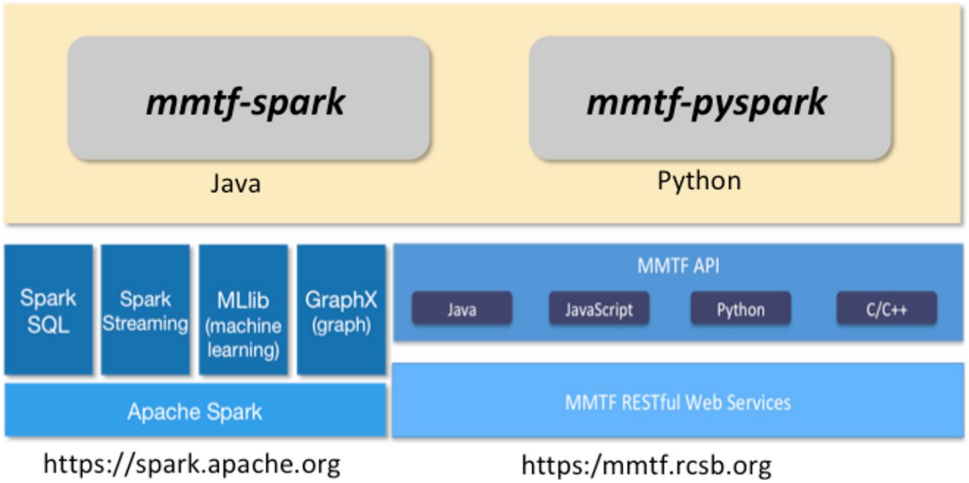
MMTF + SPARK

mmtf-spark (Java)

- High performance processing
- Suitable for large-scale calculations
- Integration with other libraries, e.g., BioJava

mmtf-pyspark (Python)

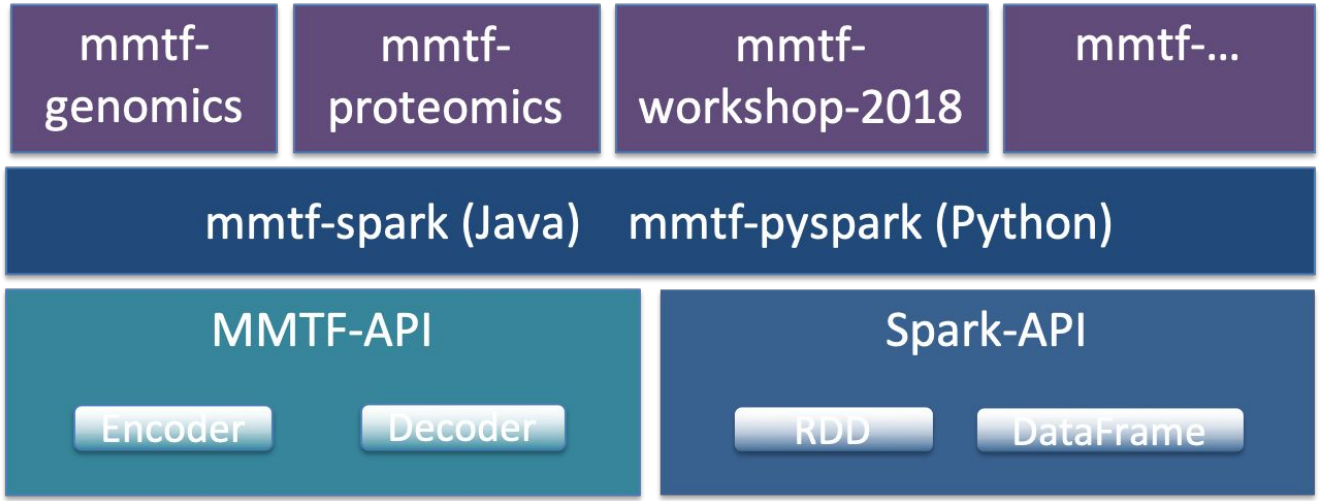
- Interactive scripting
- 2D and 3D visualization
- ML/DL tool ecosystem
- Sharable data analysis in Jupyter Notebooks



Scalable Processing of 3D Structures

MMTF (Macromolecular Transmission Format) for compact data storage, transmission, and high-performance parsing.

PDB is downloadable as an MMTF Hadoop Sequence file (**~9.3 GB for 150K structures**) for parallel processing (https://mmtf.rcsb.org)



https://github.com/sbl-sdsc

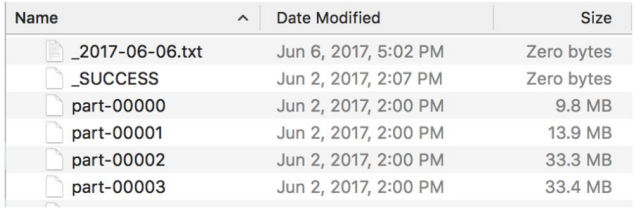
Hadoop “Sequence” Files

- **A flat file of binary key/value pairs**
- **Used by Big Data Frameworks (Hadoop, Spark)**
  - File systems need few big files for efficient processing
- **Files are splittable**
  - Can be processed in parallel
- **Often consists of a directory of Sequence files**
- **See https://wiki.apache.org/hadoop/SequenceFile**

MMTF Hadoop Sequence Files

- **Two representations**
  - **full**
    - all atoms
    - full data precision
  - **reduced**
    - polymers
      - polypeptides: C-alpha
      - polynucleotides: P
      - 1st model only (e.g., NMR)
      - no alternative locations
      - except polysaccharides
        - » all atom
    - non-polymers
      - all atoms
    - water
      - excluded
    - Reduced precision (0.1): coordinates, temperature-factor, occupancy

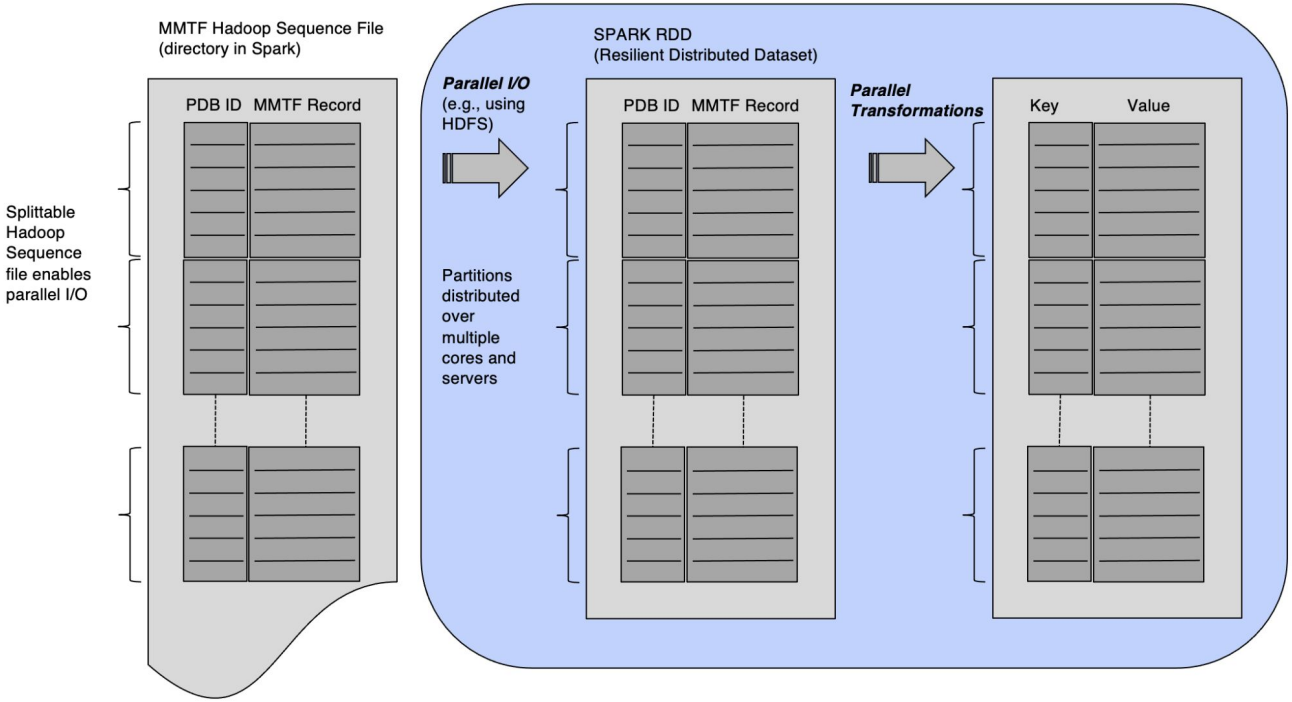
- **Example: full directory structure**



Name	Date Modified	Size
_2017-06-06.txt	Jun 6, 2017, 5:02 PM	Zero bytes
_SUCCESS	Jun 2, 2017, 2:07 PM	Zero bytes
part-00000	Jun 2, 2017, 2:00 PM	9.8 MB
part-00001	Jun 2, 2017, 2:00 PM	13.9 MB
part-00002	Jun 2, 2017, 2:00 PM	33.3 MB
part-00003	Jun 2, 2017, 2:00 PM	33.4 MB

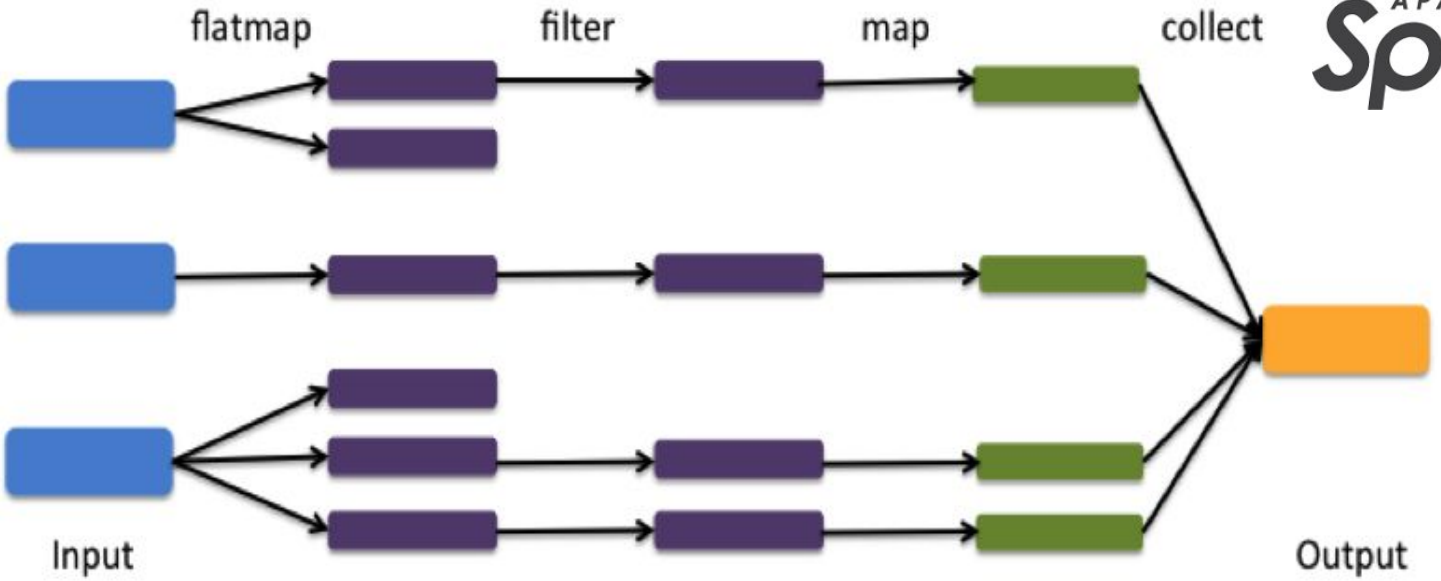
- **Timestamp file (release date)**
  - _yyyy-mm-dd.txt
- **Updated every Wed. ~00:00 UTC**
- **Multiple sequence files**
  - part-00000 …
- **Download**
  - https://mmtf.rcsb.org/download.html

MMTF Spark Parallel Data Pipeline



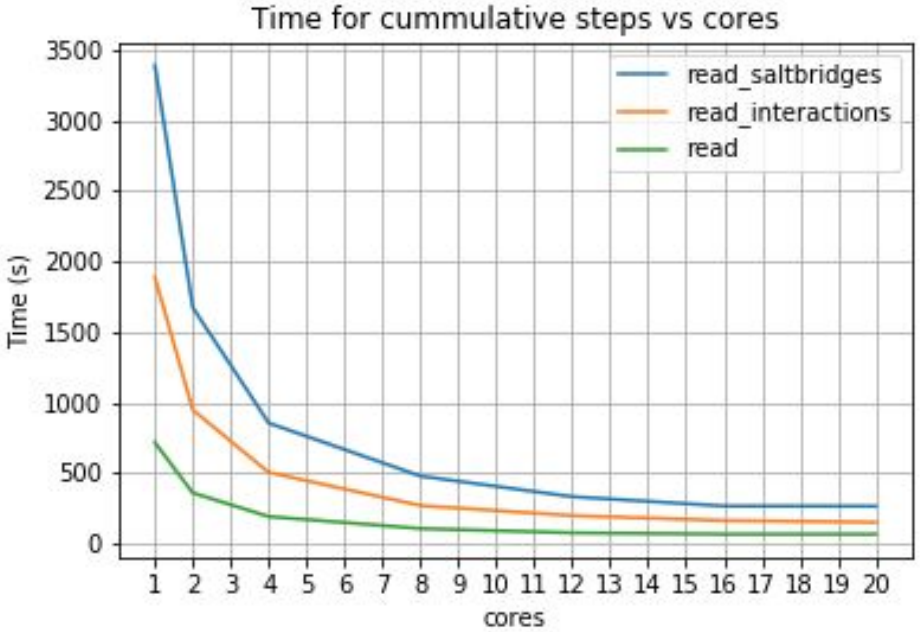
Demo:
https://github.com/sbl-sdsc/mmtf-workshop-2018
(show Activity Monitor)

Scaling Calculations with Spark



Resilient Distributed Datasets (RDD)

Parallel Processing of the PDB with mmtf-pyspark



Benchmark for (a) reading the PDB archive (148,800 structures), (b) reading PDB + finding zinc interactions, and (c) reading PDB + finding salt-bridges. The benchmark was run on a VM with 12 physical cores (Intel Xeon CPU E5-2650 0 @ 2.00GHz, hyperthreading enabled) at CyVerse. Beyond the 12 physical cores, the calculations become I/O-bound.

Enable Interactive Exploration

MMTF-PySpark in Jupyter Notebook

Interactive and reproducible data mining of the Protein Data Bank
-> access to Python ecosystem of data analysis and ML tools

Read PDB and create PISCES non-redundant set

```
pdb = MmtfReader.readSequenceFile(path, sc)
pdb = pdb.filter(pisces(sequenceIdentity = 20, resolution = 2.0))
```

RDD

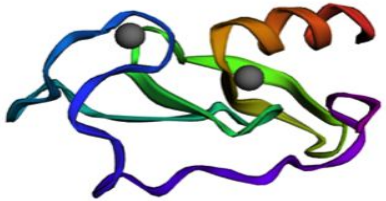
Extract Zinc interactions

```
finder = groupInteractionExtractor("ZN", distance = 3.0)
interactions = finder.getDataset(pdb)
```

Dataframe

Visualize first hit

```
hit = interactions.first()[0]
view = py3Dmol.view(query='pdb:%s'%hit)
view.setStyle({'cartoon': {'color':'spectrum'}})
view.setStyle({'atom':'ZN'},{'sphere': {'color':'gray'}})
view.show()
```



Show top 5 interacting groups

```
interactions.filter("element2 != 'C'").groupBy("residue2")
            .count().sort("count", ascending=False).show(5)
```

```
+--------+-----+
|residue2|count|
+--------+-----+
|     CYS| 1394|
|     HIS| 1265|
|     HOH| 1049|
|     GLU|  737|
|     ASP|  722|
+--------+-----+
```

Demos of some Core Operations

- **Filtering**

https://github.com/sbl-sdsc/mmtf-workshop-2018/blob/master/3-mmtf-pyspark/2-Filtering.ipynb

- **Flatmapping**

https://github.com/sbl-sdsc/mmtf-workshop-2018/blob/master/3-mmtf-pyspark/4-Flatmapping.ipynb

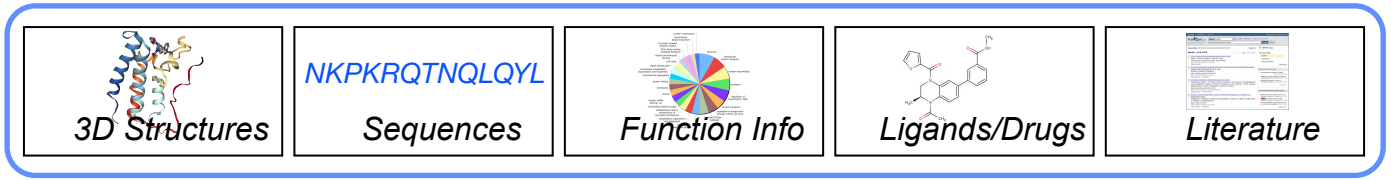
- **Map/Reduce**

https://github.com/sbl-sdsc/mmtf-workshop-2018/blob/master/3-mmtf-pyspark/5-MapReduce.ipynb

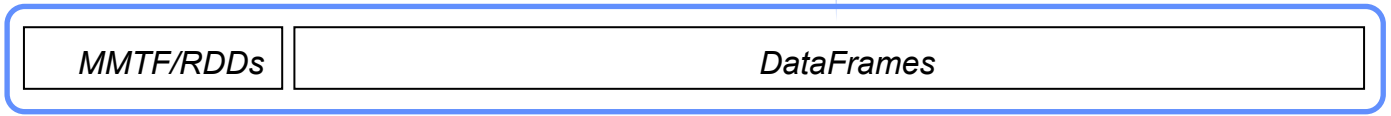
Working with Dataframes

Integrating Diverse Bio Resources

Diverse data sources



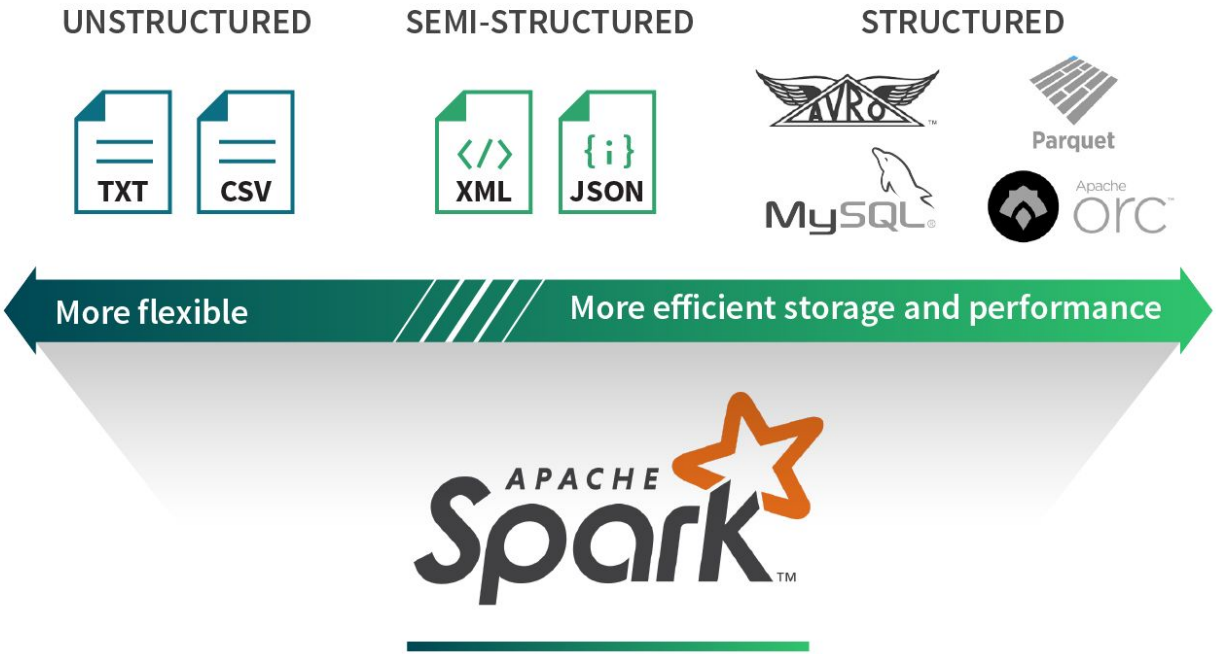
Standardized, distributed data structures for parallel processing



Applications

- *3D Structural analysis, clustering, visualization*
- *Data integration and calculation of derived data*
- *Creation of datasets for machine learning*
- *Machine/deep learning*
- *Natural language processing*

Data Formats for Spark



https://databricks.com/blog/2017/02/23/working-complex-data-formats-structured-streaming-apache-spark-2-1.html

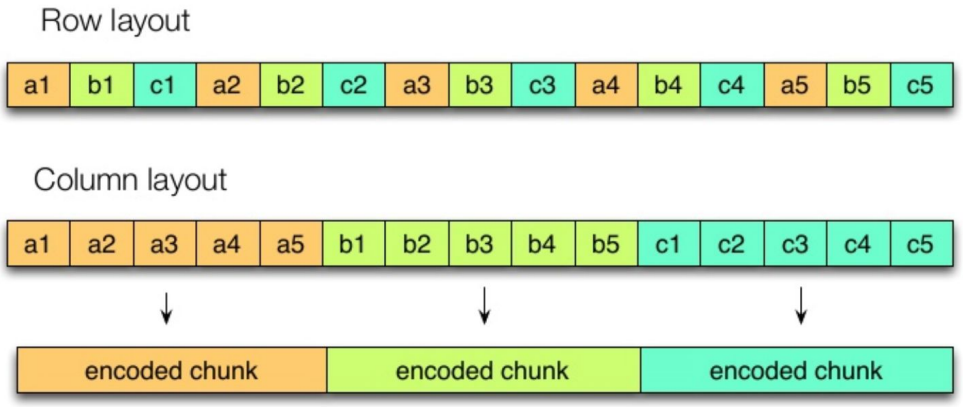
Columnar Storage (Parquet, ORC)

- Space efficiency
- Query performance
  - Predicate pushdown
  - Indexing

Nested schema

Logical table representation

a	b	c
a1	b1	c1
a2	b2	c2
a3	b3	c3
a4	b4	c4
a5	b5	c5



@EmrgencyKittens

https://www.slideshare.net/julienledem/if-you-have-your-own-columnar-format-stop-now-and-use-parquet

Benchmark: Storage Formats & Query Performance

UniProt sequence to PDB residue level mappings obtained from the PDBe SIFTS project (Structure Integration with Function, Taxonomy and Sequence).

Downloading and parsing of the original ~140,000 xml.gz files took 27 hours. This dataset contains **105 million records**.

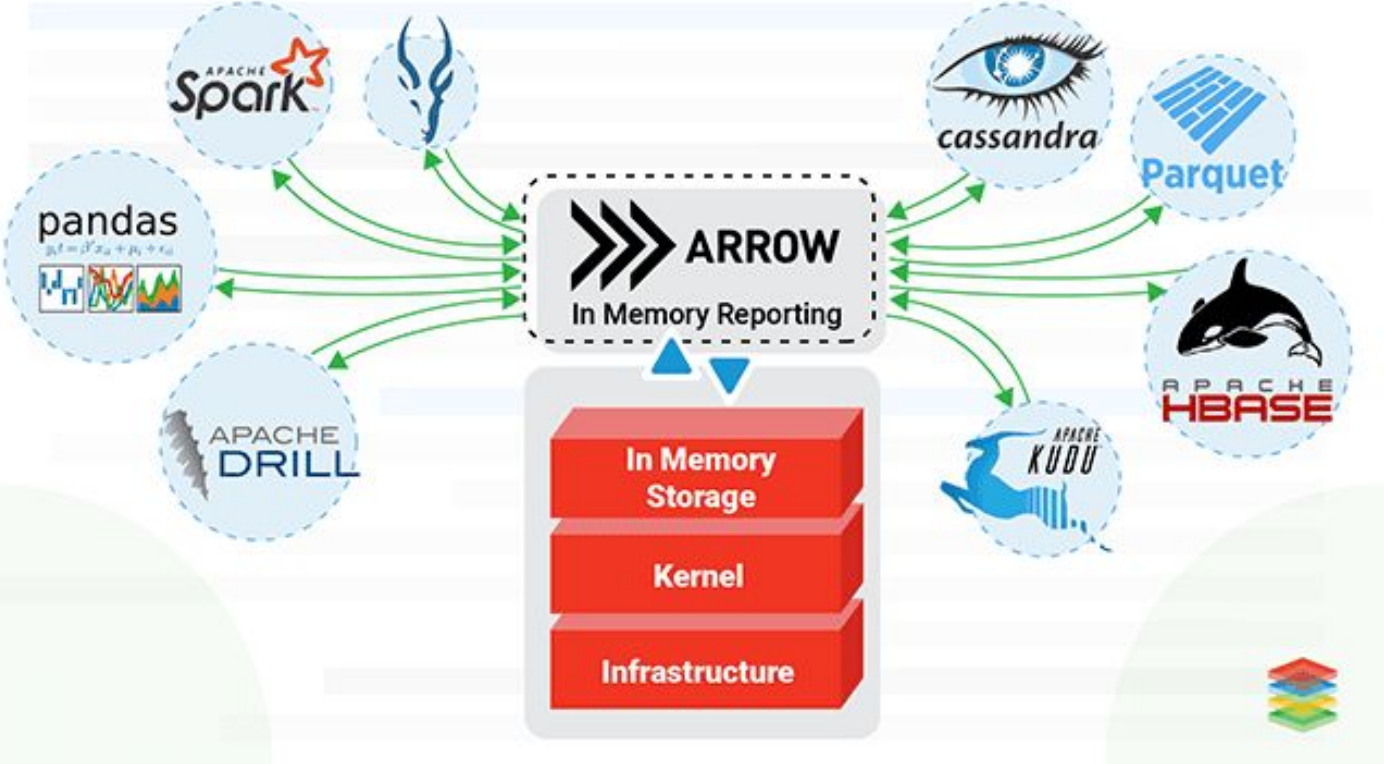
Dataset name	File format	Compression codec	Size (MB)
xml_gzip	xml	gzip	~5200
csv_gzip	csv	gzip	519.7
parquet_snappy	parquet	snappy	145.1
parquet_gzip	parquet	gzip	**57.9**
orc_zlib	orc	zlib	41.9
orc_lzo	orc	lzo	**41.7**

Dataset operations, timing includes reading data from disk (Mac Pro, 2 cores, SSD)

Benchmark	orc_lzo (second)	parquet_gzip (seconds)
Count	**3.7**	4.1
Query	**11.9**	20.2
Join	**12.0**	23.3
Convert	**6.0**	7.9

https://github.com/sbl-sdsc/sifts-columnar

In-Memory Transfer with Apache Arrow



Deploying SPARK Applications on Public Infrastructure

Reproducible Data Analysis in Jupyter Notebooks

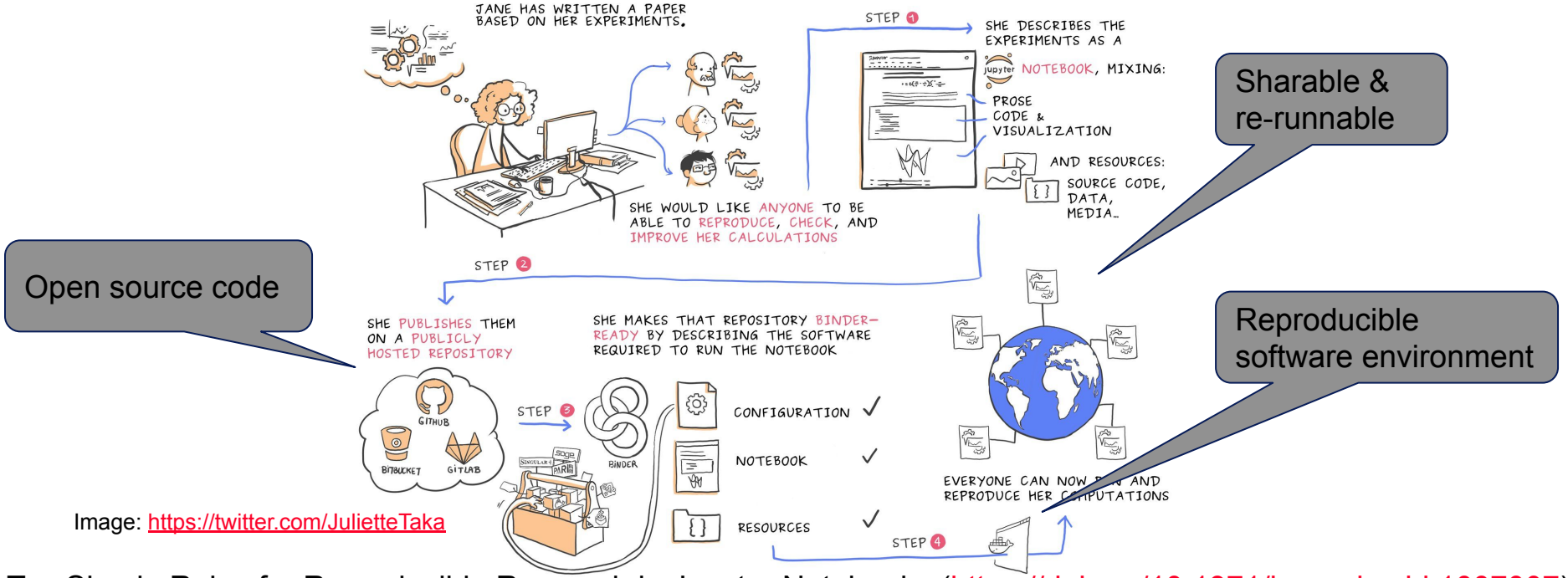


Image: https://twitter.com/JulietteTaka

Ten Simple Rules for Reproducible Research in Jupyter Notebooks (https://doi.org/10.1371/journal.pcbi.1007007)

Binder Setup Example

- **Conda environment**

https://github.com/sbl-sdsc/mmtf-genomics/blob/master/binder/environment.yml

- **Setup SPARK configuration**

https://github.com/sbl-sdsc/mmtf-genomics/blob/master/binder/start

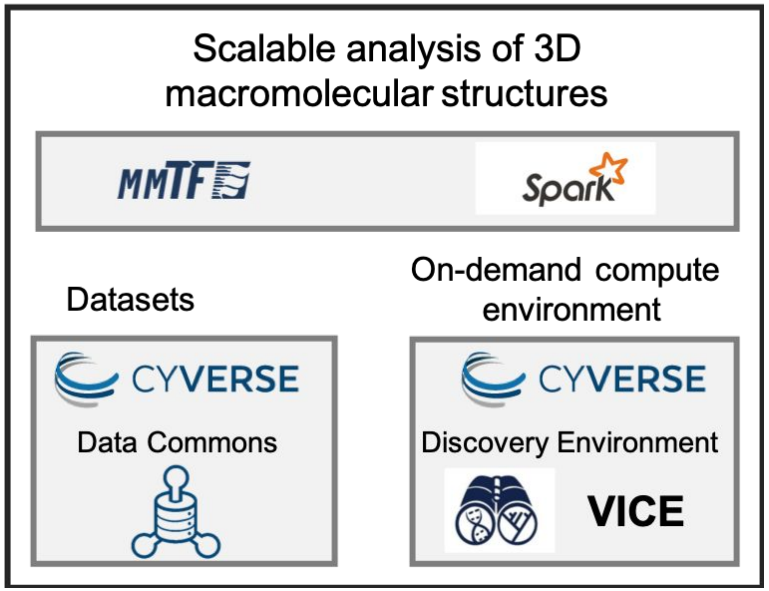
- **Binder launch link in README.md**

[![Binder](https://binder.pangeo.io/badge_logo.svg)](https://binder.pangeo.io/v2/gh/sbl-sdsc/mmtf-genomics/master?urlpath=lab)

See tutorial: A Practical Introduction to Reproducible Computational Workflows

https://github.com/ISMB-ECCB-2019-Tutorial-AM4/reproducible-computational-workflows

CYVERSE Setup Example



https://de.cyverse.org/de/

- Free user account
- 100 GB of storage
- More cores/memory
- Requires a Dockerfile

Example Dockerfile
https://github.com/sbl-sdsc/mmtf-genomics/tree/master/vice

mmtf-genomics Demo

https://github.com/sbl-sdsc/mmtf-genomics

- Set of Jupyter Notebooks to map mutations onto 3D protein structures
- **Demo:**
  - Map mutations in the SARS-CoV-2 Spike protein
  - Use COVID-19-Net Knowledge Graph to get variant information for about 1 million strains from genome sequencing (https://github.com/covid-19-net/covid-19-community)
  - Use mmtf-pyspark to calculate interactions between the Spike protein and Antibodies

When to use SPARK

- PySpark is complex
  - Python <- Py4J/Arrow ->SCALA/JVM
  - cryptic error messages
  - difficult to debug
- If data fits into memory -> use Pandas
- If you want a pure Python solution -> use Dask
- For interactive applications use standalone mode on a single server (many cores, lots of memory)
- For deployment on a cluster you need a cluster manager such as YARN or Kubernetes

Acknowledgements

RCSB PDB, PDBe, PDBj teams

mmtf-pyspark
Mars Huang, SDSC

dbSNP data integration
Lon Phan, NIH/NLM/NCBI

Genome to Structure mapping
Jianjong Gao, Memorial Sloan Kettering Cancer Center
Juexin Wang, U. Missouri

CyVerse
Nirav Merchant
Upendra Devisetty
Ian Mian
and other team members

Funding: NCI/NIH Award Number U01CA198942 and SDSC