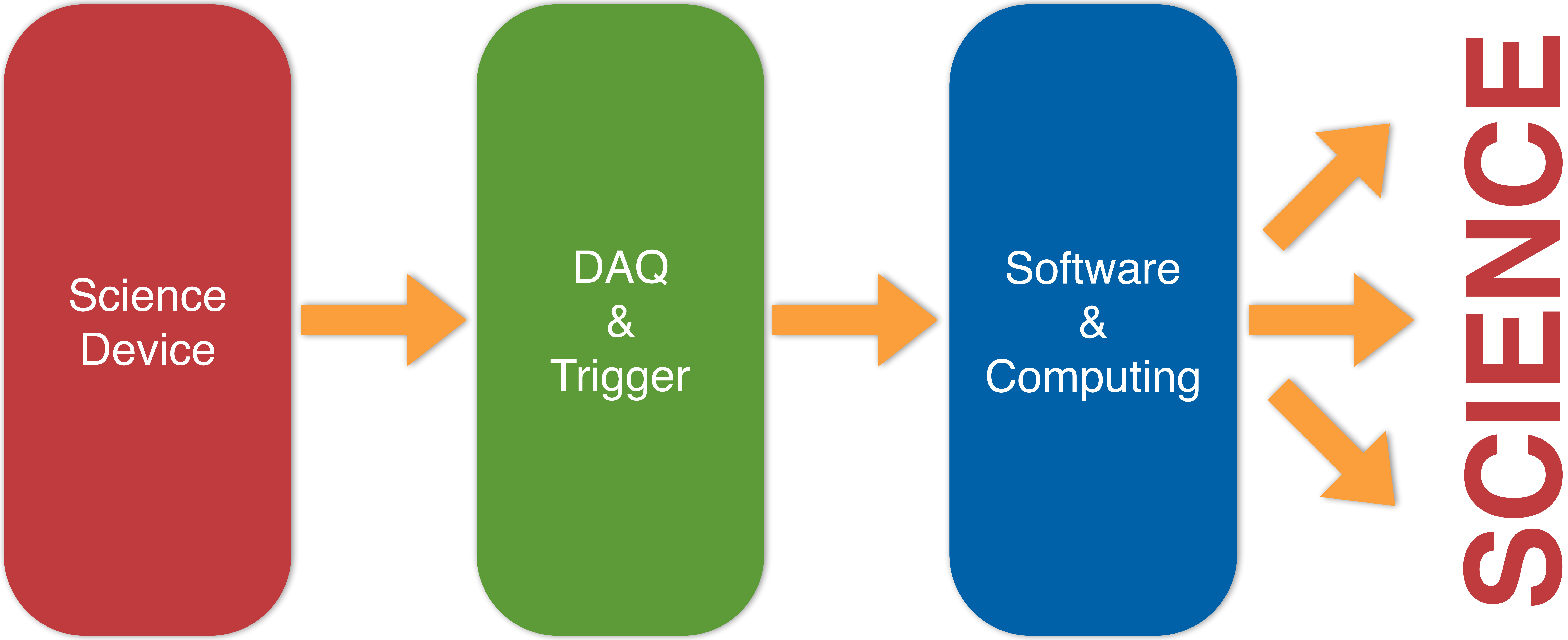
Status and Plans of the CMS Big Data Project

Oliver Gutsche, Matteo Cremonesi, Bo Jayatilaka, Jim Kowalkowski, Saba Sehrish - Fermi National Accelerator Laboratory
Peter Elmer, Jim Pivarski, Alexey Svyatkovskiy - Princeton University
Maria Girone, Luca Canali, Kacper Surdy, Vaggelis Motesnitsalis - CERN
Ian Fisk - Simons Foundations
Viktor Khristenko - University of Iowa

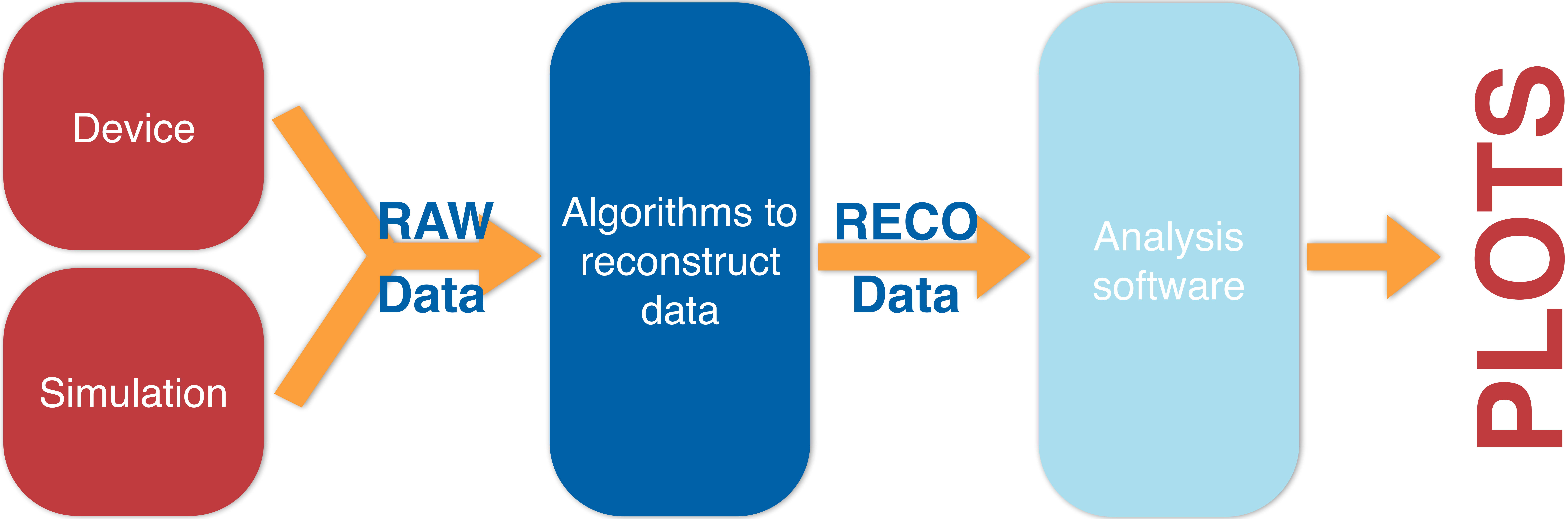
CERN Database Futures Workshop, 29. May 2017

The Scientific Process



- Software & Computing is an integral part of the scientific process

Software & Computing



- **Software** is important for every step on the way to scientific results

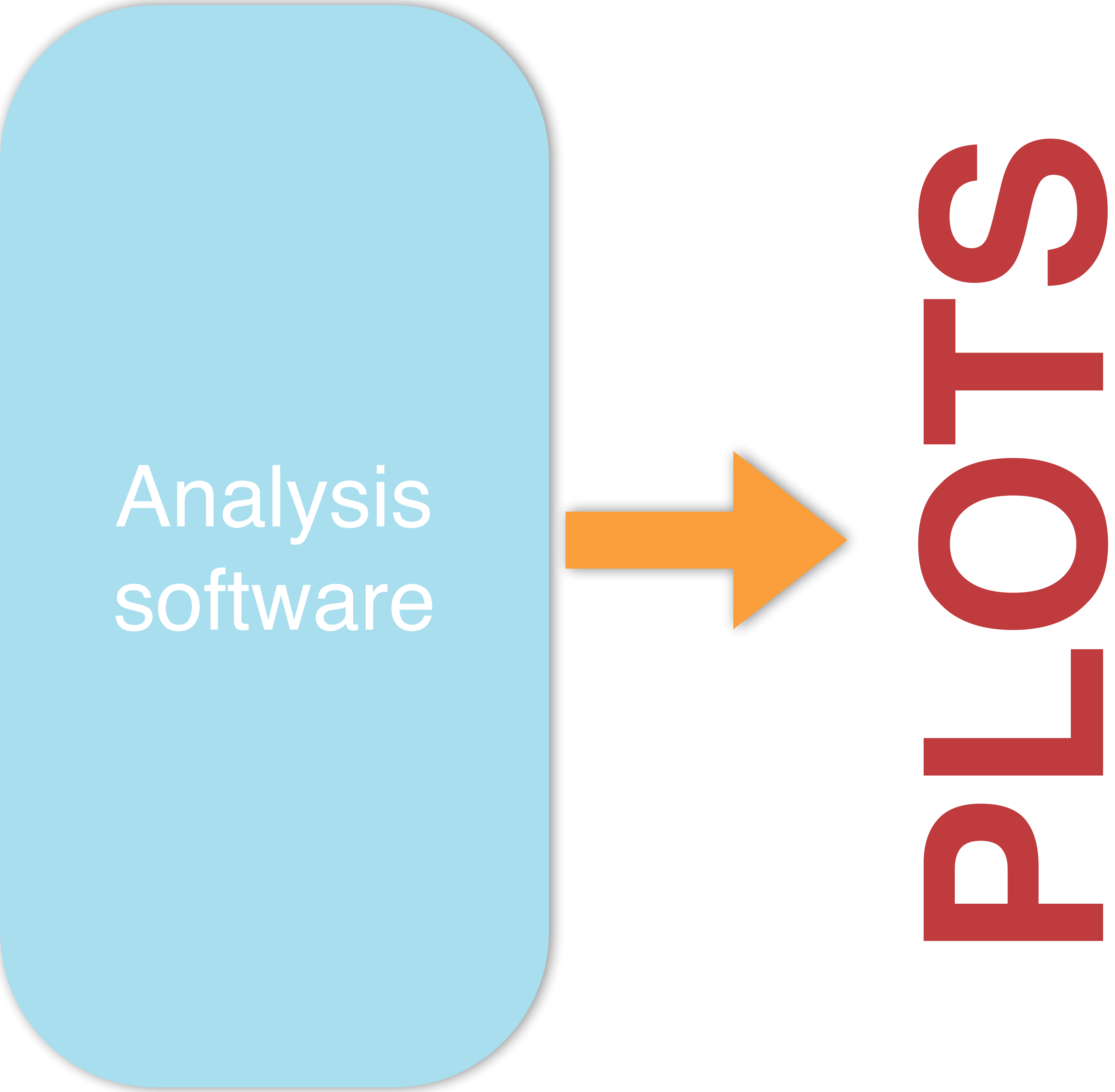
Software & Computing

- **Central:** organized processing/production, one software stack/framework per experiment (C++), one or few output sets, shared by large parts of the experiments for analysis
- **Chaotic:** smaller groups down to individuals explore the data for analysis, individually implemented analysis code, because of data volumes

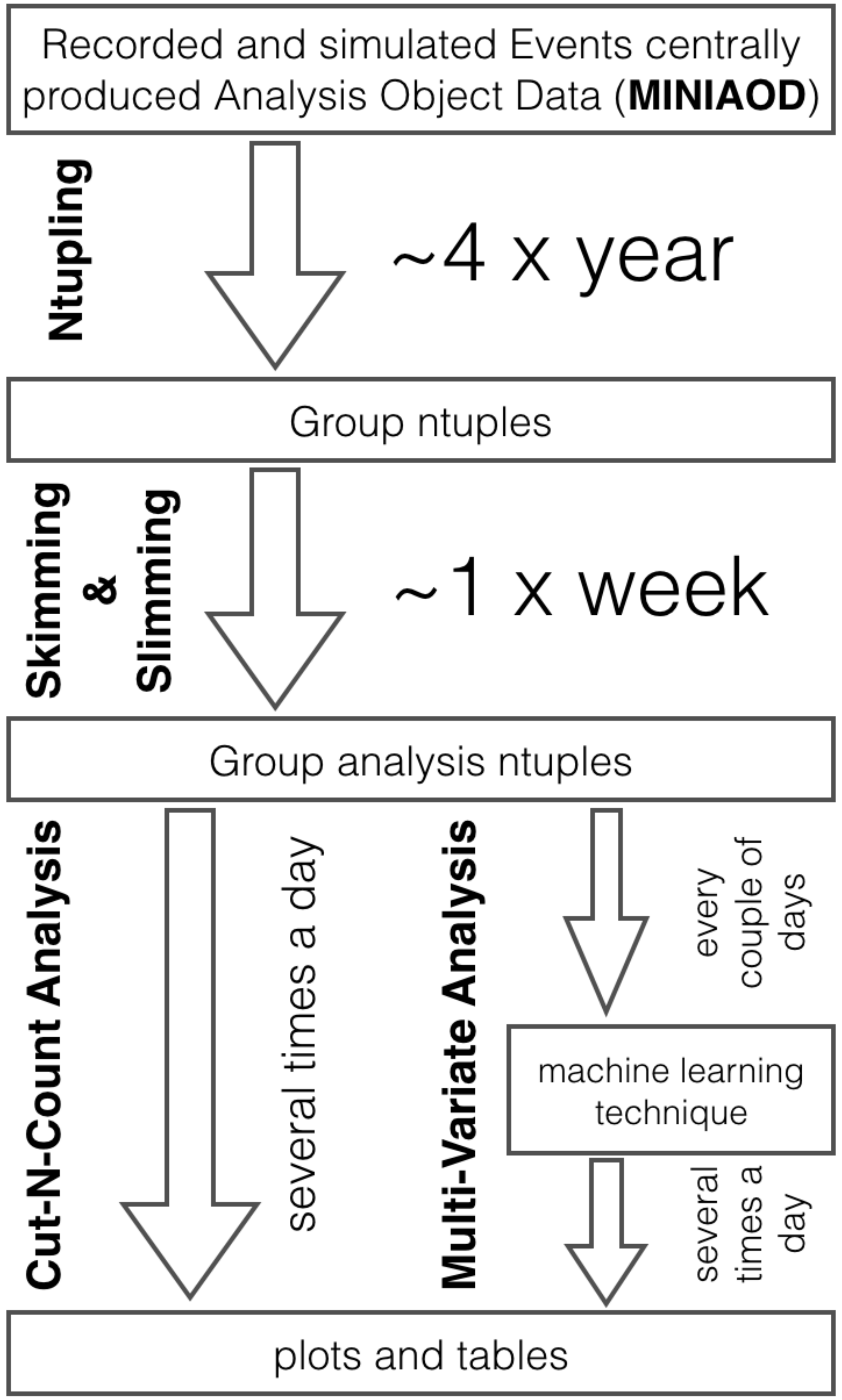
Physics Data Reduction

- Data analysis needs fast turn-around
  - “Interactivity” is a big need for efficient data exploration
- Each analysis is different
  - looks for different physics
- Data volumes will reach multi-PB sizes in the future
  - input data composition different for every analysis
- Analysts need to reduce data to be able to analyze it (or do they?)

- Requirements:
  - Reduce data by skimming (filter specific collisions) and slimming (reduce content per collision)
  - Calculate new properties before skimming and slimming
  - Re-calculate properties previously calculated centrally
  - All this by multiple users in parallel



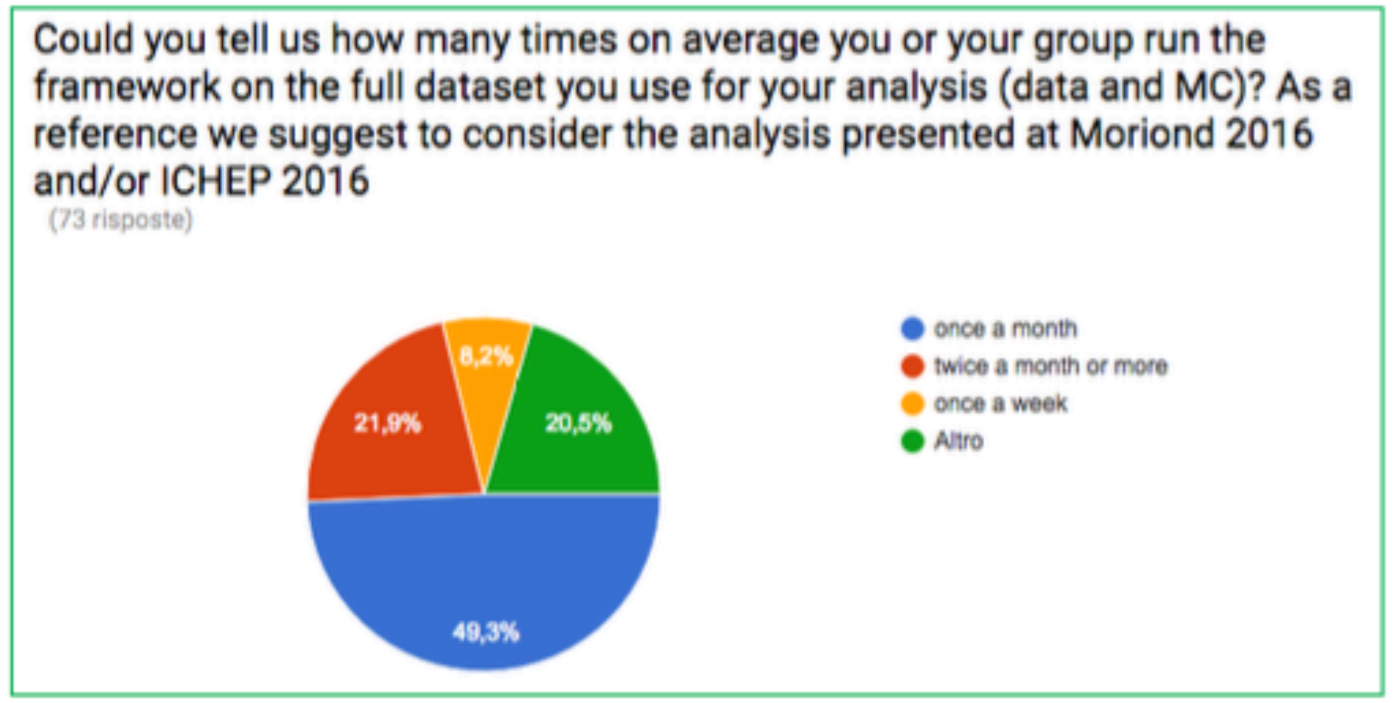
Analysis - A multi-step process



- Current Analysis Workflow
  - Touches only a subset of the total data volume, but subset varies from analysis to analysis
  - Complicated multi-step workflow because dataset is too large for interactive analysis
    - Slimming & Skimming, analysis dependent
    - Calculation of new quantities
    - Rerun framework code (b-tagging with non-default parameters, etc.)
    - Recipes on top of centrally produced samples to correct problems/mistakes
  - Can take weeks using GRID resources and local batch systems
    - Experiments started to centralize first step
  - Not all time spent is actual CPU, a lot of time is bookkeeping, resubmission of failed jobs, etc.
- Currently based on ROOT
  - ROOT is the community's statistics, plotting and I/O toolkit
  - Developed by the community and optimized for this purpose

CMS User Analysis Survey

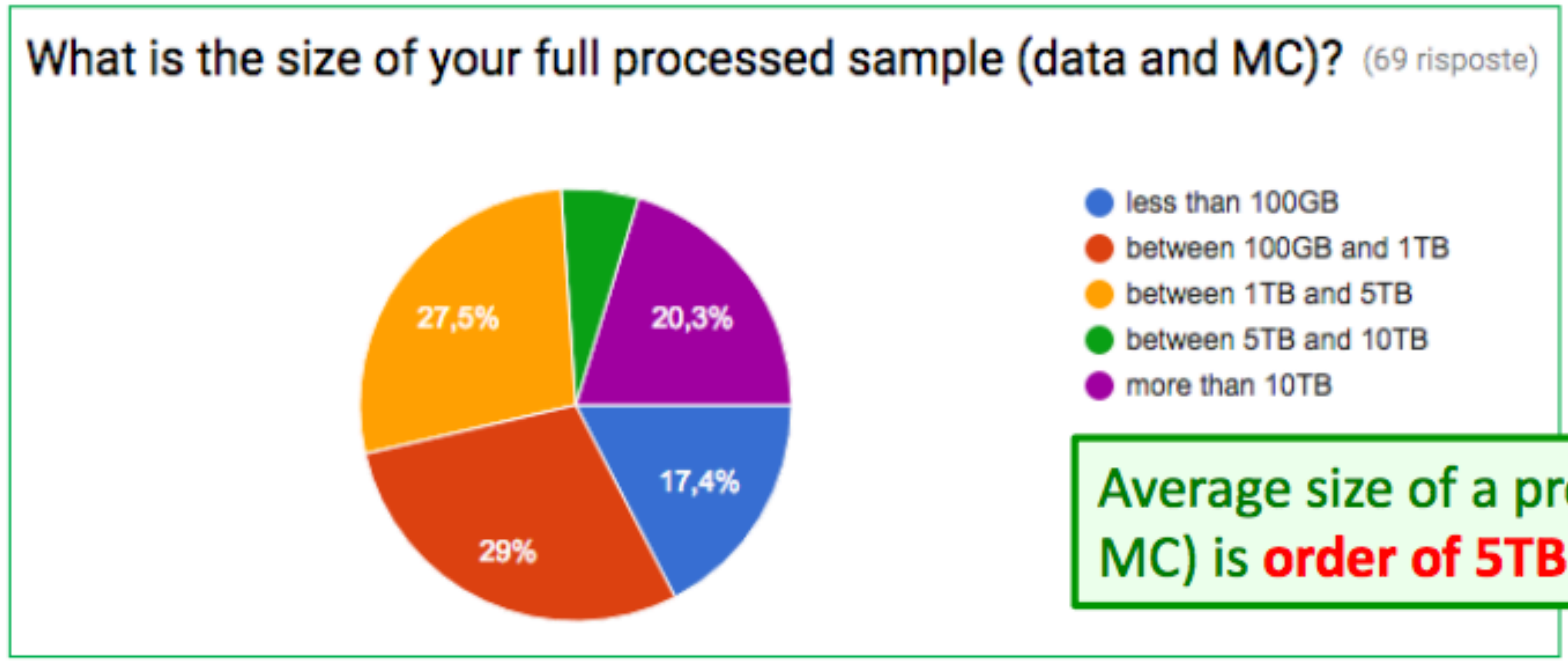
- Main outcomes
  - 85% of the users use an independent Framework…
  - Almost all users mention 1 or 2 intermediate steps to produce User/Group specific root trees
    - At least 40% of the answers mention the word flat trees
    - Counted around 40 different FW used within the CMS community
  - More than 35% of the answers mention the words skim/ reducing



Hard to quantify (I cannot check how many users refers to the same FW) but…

It seems that **each year** the CMS computing infrastructure is used **order of 1000 times** to run on a full dataset (data and MC)

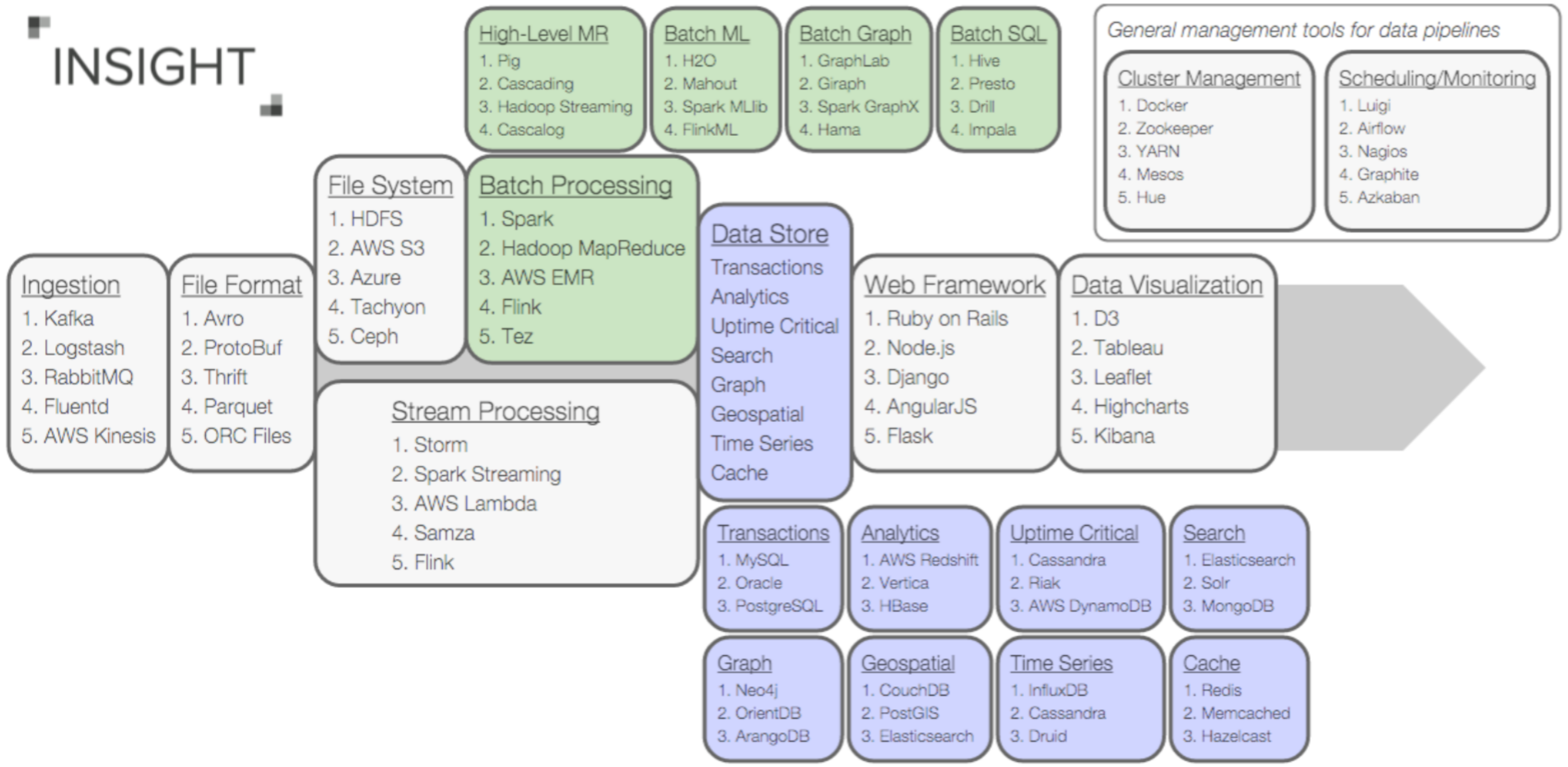
Main reasons to rerun are **POG updates** and adding **new variables**



Average size of a processed dataset (data and MC) is **order of 5TB** but it could much bigger

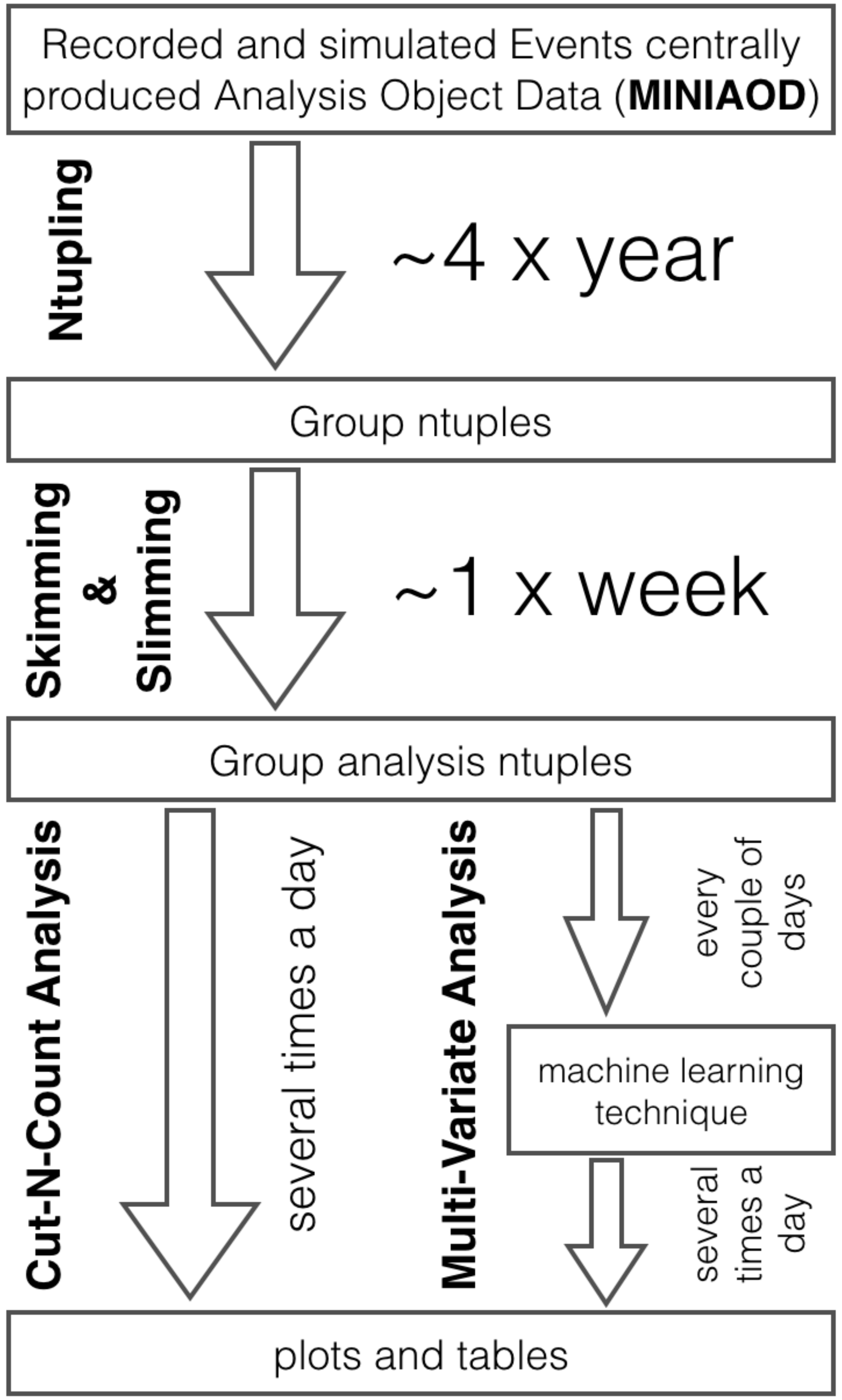
Big Data

- New toolkits and systems collectively called “Big Data” technologies have emerged to support the analysis of PB and EB datasets in industry.



- Our goals in applying these technologies to the HEP analysis challenge:
  - Reduce time-to-physics
  - Educate our graduate students and post docs to use industry-based technologies
    - Improves chances on the job market outside academia
    - Increases the attractiveness of our field
  - Be part of an even larger community

Feasibility Studies: Two Thrusts



- Input: MINIAOD
  - Caveat: Applying recipes or re-running framework code currently not being considered
- Thrust 1:
  - Use analysis-specific data formats that have all recipes applied and framework code re-run
  - Explore using Apache spark producing plots and tables
- Thrust 2:
  - Use official input
  - Demonstrate reduction capabilities producing group analysis ntuples
    - Goal: reduce 1 PB input to 1 TB output in 5 hours
  - Intel CERN Openlab project

New Tools

DIANA: Histogrammar

- http://histogrammar.org
- The ROOT histogram API is intended to be used in a **user-controlled event loop**, which **isn't available in Spark** because Spark manages concurrency for you.
- Histogrammar was designed to be a better fit to this sort of environment because it additionally provides a functional interface:
  - You fill histograms by passing lambda functions, in the same way that you perform transformations in Spark.
  - Filled Histogrammar histograms can be immediately converted to ROOT for further processing.
  - Analysis code is now independent of where the data are analyzed.
- Side effect: moving the logic of data analysis out of the for loop allows the analyst to describe an entire analysis declaratively.

histo·*grammar*
/histō,ˈɡɹæm.ər/

MAKING HISTOGRAMS FUNCTIONAL

ROOT:

```
histogram = ROOT.TH1F("name", "title", 100, 0, 10)
for muon in muons:
    if muon.pt > 10:
        histogram.fill(muon.mass)
```

Histogrammar:

```
histogram = Select(lambda mu: mu.pt > 10,
                   Bin(100, 0, 10, lambda mu: mu.mass,
                       Count()))
for muon in muons:
    histogram.fill(muon)
```

DIANA: spark-root

- Read ROOT files directly from Apache Spark
  - Connect ROOT to ApacheSpark to be able to read ROOT TTrees, infer the schema and manipulate the data via Spark's DataFrames/ Datasets/RDDs.
- https://github.com/diana-hep/spark-root

```
df = sqlContext.read.format("org.dianahep.sparkroot").option("tree", "Events").load("hdfs:/cms/big
datasci/vkhriste/data/publiccms_muionia_aod")
#df1 = sqlContext.read.format("org.dianahep.sparkroot").option("tree", "Events").load("hdfs:/cms/b
igdatasci/vkhriste/data/publiccms_muionia_aod/0000/FEEFB039-0978-E011-BB60-E41F131815BC.root")
df.printSchema()
```

```
root
 |-- EventAuxiliary: struct (nullable = true)
 |    |-- processHistoryID_: struct (nullable = true)
 |    |    |-- hash_: string (nullable = true)
 |    |-- id_: struct (nullable = true)
 |    |    |-- run_: integer (nullable = true)
 |    |    |-- luminosityBlock_: integer (nullable = true)
 |    |    |-- event_: integer (nullable = true)
 |    |-- processGUID_: string (nullable = true)
 |    |-- time_: struct (nullable = true)
 |    |    |-- timeLow_: integer (nullable = true)
 |    |    |-- timeHigh_: integer (nullable = true)
 |    |-- luminosityBlock_: integer (nullable = true)
 |    |-- isRealData_: boolean (nullable = true)
 |    |-- experimentType_: integer (nullable = true)
 |    |-- bunchCrossing_: integer (nullable = true)
 |    |-- orbitNumber_: integer (nullable = true)
 |    |-- storeNumber_: integer (nullable = true)
 |-- EventBranchEntryInfo: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- branchID_: struct (nullable = true)
 |    |    |    |-- id_: integer (nullable = true)
 |    |    |-- productStatus_: byte (nullable = true)
 |    |    |-- parentageID_: struct (nullable = true)
 |    |    |    |-- hash_: string (nullable = true)
 |    |    |-- transients_: struct (nullable = true)
 |-- EventSelections: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- hash_: string (nullable = true)
 |-- BranchListIndexes: array (nullable = true)
 |    |-- element: short (containsNull = true)
 |-- L1GlobalTriggerObjectMapRecord_hltL1GtObjectMap__HLT_: struct (nullable = true)
 |    |-- edm::EDProduct: struct (nullable = true)
```

```
In [6]: df.count()
Out[6]: 12058887

In [7]: slimmedEvents = df.select("recoMuons_muons__RECO_.recoMuons_muons__RECO_obj.reco::RecoCandidate.re
co::LeafCandidate")

slimmedEvents.show()

+--------------------+
| reco::LeafCandidate|
+--------------------+
|[[[],-3,3.085807,...|
|[[[],3,4.1558356,...|
```

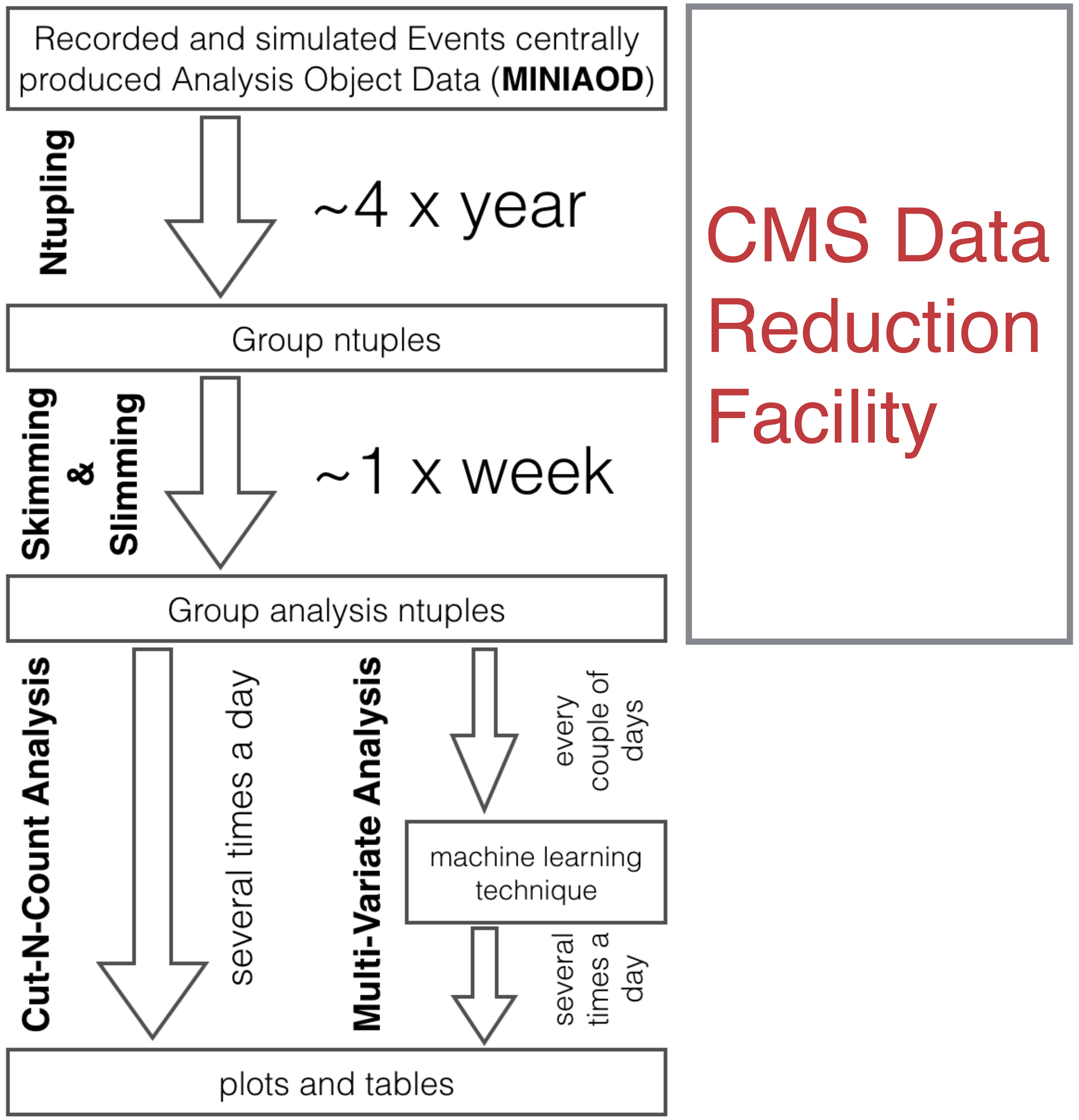
Thrust 1: Usability Study

Thrust 1: Usability Study - Status

- CHEP 2016 paper accepted for publication
  - https://arxiv.org/abs/1703.04171
- Study based on monoTop Dark Matter analysis
  - Conversion to AVRO format and upload to HDFS
  - Analysis implemented in Scala
  - Processing in Apache Spark
  - Result:
    - Spark analysis simpler to structure (functional programming) and easier to port
    - Performance comparison challenging (apples-to-apples comparison)
- Next steps
  - New analysis framework for monoTop
  - Use ROOT files directly in Spark
  - Use analysis code in Scala and use Histogrammar
  - Achieve apples-to-apples comparison to ROOT analysis

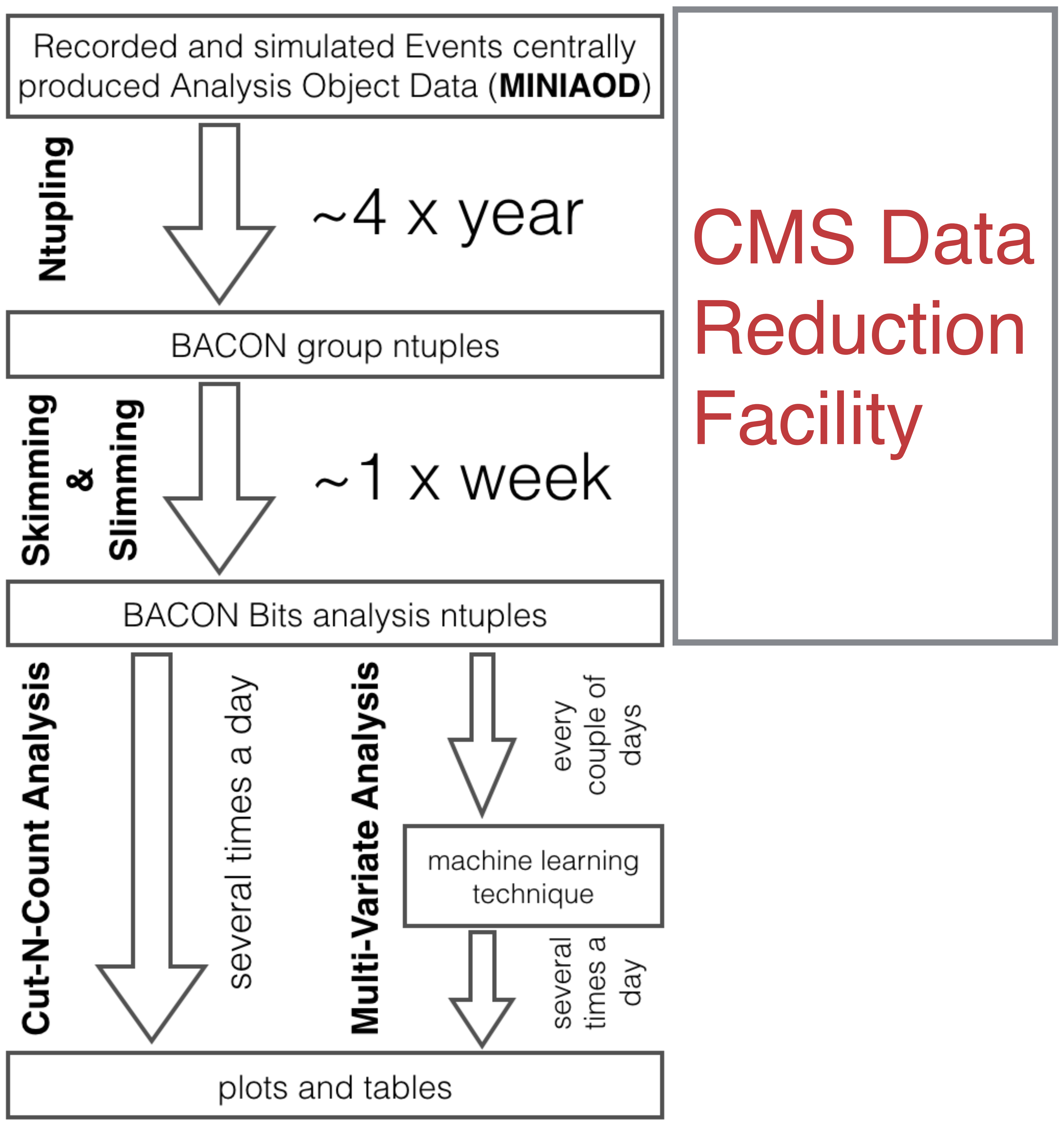
CERN Openlab/Intel CMS Data Reduction Facility Project

CMS Data Reduction Facility



- CERN Openlab project with Intel (2 years)
- Demonstration facility optimized to read through petabyte sized storage volumes
  - Produce sample of reduced data based on potentially complicated user queries
  - Time scale of hours and not weeks as it currently requires.
- If successful, this type of facility could be a big shift in how effort and time is used in physics analysis
  - Same infrastructure and techniques should be applicable to many sciences

Project Objectives



- We would like to demonstrate the ability to reach **at least a 1000 fold reduction** in selected data
- We would like to show that with an optimized prototype center that we can perform this task **roughly 100 times faster** than it can currently be done
- Goal:
  - Process an input sample of 1PB within 5 hours
  - Export a selected sample that is at least 1000 times smaller

Thrust 2: CMS Data Reduction Facility

Thrust 2: CMS Data Reduction Facility - Status

- Intel/CERN fellow started at CERN in March 2017
  - Welcome Vaggelis!
- Work on CERN Hadoop using Spark
  - Enable Spark to read ROOT files through spark-root directly from EOS (Vaggelis)
- Started with using CMS open data
  - Copied small amounts to HDFS (currently using 1.2 TB)
- Next steps
  - Start with CMS Open Data and execute a suitable ntuple production step with significant reduction
    - Download reduction result and make physics-style plots
  - Scale up and study performance

Conclusions & Outlook

Conclusions & Outlook

- Investigating Big Data technologies to solve the HL-LHC data analysis challenge ➜ Apache Spark as a starting point
  - Fulfills immediately 2 out of 3 goals:
    - Educates our community to use industry-based technologies
    - Uses tools developed in larger communities reaching outside of our field
  - First study accepted for publication in CHEP 2016 proceedings
- Thrust 1: Usability Study
  - Adapt to new framework, read ROOT files directly, use Histogrammar
- Thrust 2: Intel/CERN openlab CMS Data Reduction Facility
  - Use CMS open data as a starting point, read ROOT files directly from EOS, scale up and study performance


Fermilab
U.S. DEPARTMENT OF ENERGY
Office of Science