File Level Provenance Tracking in CMS

Christopher Jones
Jim Kowalkowski
Marc Paterno
Liz Sexton-Kennedy
William Tanenbaum
FNAL

Daniel S Riley
Cornell University

For the Offline and Computing Project of CMS

Outline

Provenance

CMS Data Processing Model

Recording

File Format

Controlling Size

Conclusion

Why Provenance?

We record information to understand the history of how data were produced and chosen.

Provenance information does not have to be sufficient to allow an exact replay of a process

Importance of storing provenance in output files

The large scale, highly distributed nature of production requires provenance tracking to insure trust in the data

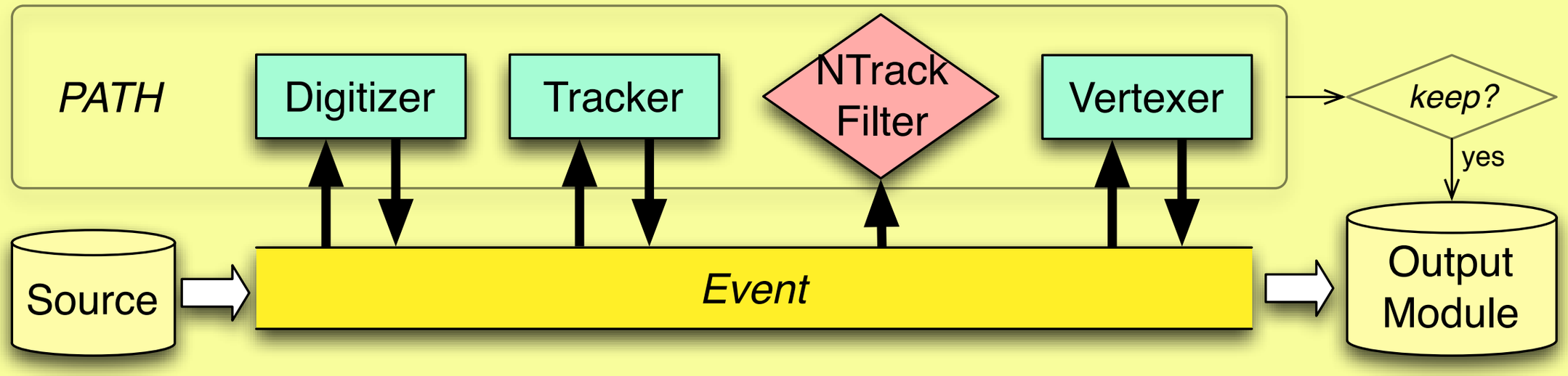
especially true for physicists personal skims which are not centrally managed

Uses

Tracking the source of a problem seen in one file but not another one
Guarantee compatibility when reading multiple files in a job
Confirm that an analysis was done using the proper data
Track why two analyses get different results

CMS' Processing Model

Processing model defines what provenance to record



Sources

Create events and read event data from previous processes

Producers

Read data from an event and push new data products into the event

Filters

Read data from an event and decide if the event passes some criteria

Paths

An ordered set of Producers and Filters where Filter decisions can cause a Path to stop

OutputModules

Look at the success/failure of a Path to decide if data from an event should be saved

Provenance Types

Process Level

What files were read and what applications were used to generate a group of files
- e.g. file DEF.root was made by application cmsRun which read file ABC.root

Handled by CMS' workflow management tools
- (see CMS workflow management talks for further details)

File Level

Record the internal state of the application to the output file

Per event level
- What filters were applied to choose the stored events
  - e.g., event filters MaxTracks and MinJets had to pass event
- What happened in the application while the event was being processed
  - e.g., module FooBar threw an exception but the framework ignored it and kept processing

Per data products level
- What Producers created which data products
  - e.g., Producer JetFinder creates a std::vector<Jet> with label 'jets'
- How were the Producers of each data product configured
  - e.g., Producer JetFinder had parameter 'threshold' set to 5

Per event per data product level
- What data products were read by the producer to create the new product
  - e.g., for event 10 the JetFinder read the list of calorimeter towers

Recording: Startup

Module construction time

Passed configuration information and register what data they produce

Defaults for configuration are injected if not already specified

Can not be externally altered after construction

Path construction time

Once constructed a Path specification can not be altered

OutputModule construction time

Last to be constructed

Records

- What Paths the OutputModule is monitoring
- Software version used in the job
- Path and Module configuration for this job

Recording: Event

Processing steps

For each processing step that added data to this event (e.g. HLT, RECO) record

- the order of the processing step
- the configuration used for that step

Producer

Monitor each data object requested by a Producer via the Event

If Producer succeeds then

- publish products it constructed
- record what data requested

Path

Record if a Path succeeds or fails

If fails record what was the last module executed and why it stopped the job

- e.g. because the filter rejected the event or because it threw an exception

When all Paths finished store information for all Paths into one object which is added to the Event

OutputModule

Run only after all Paths have finished

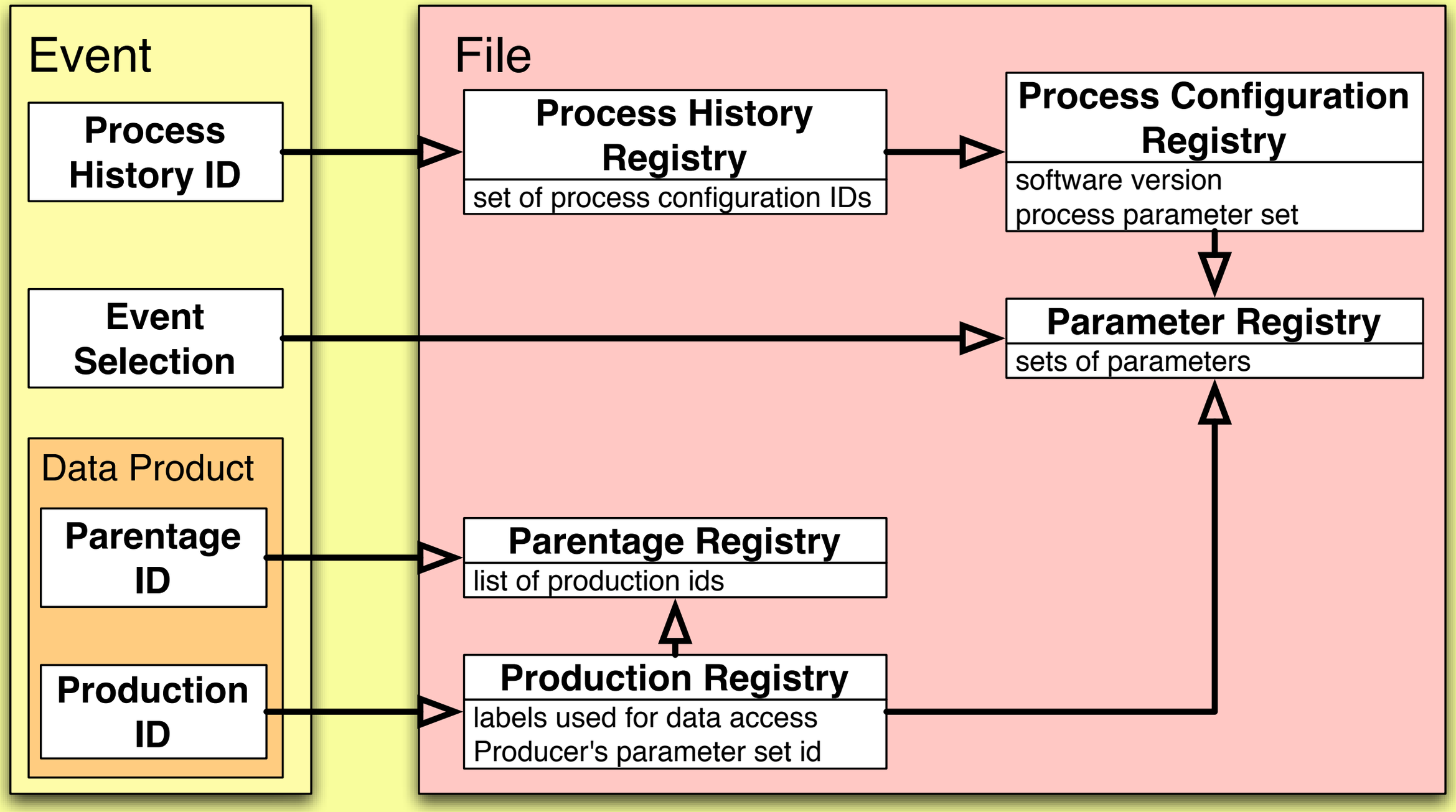
Write requested provenance for event into output

- can include provenance from previous processes copied from source

File Format

Store provenance per event and per file based on rate of change

Per event data often repeated so store info in registry and then just save ID in event

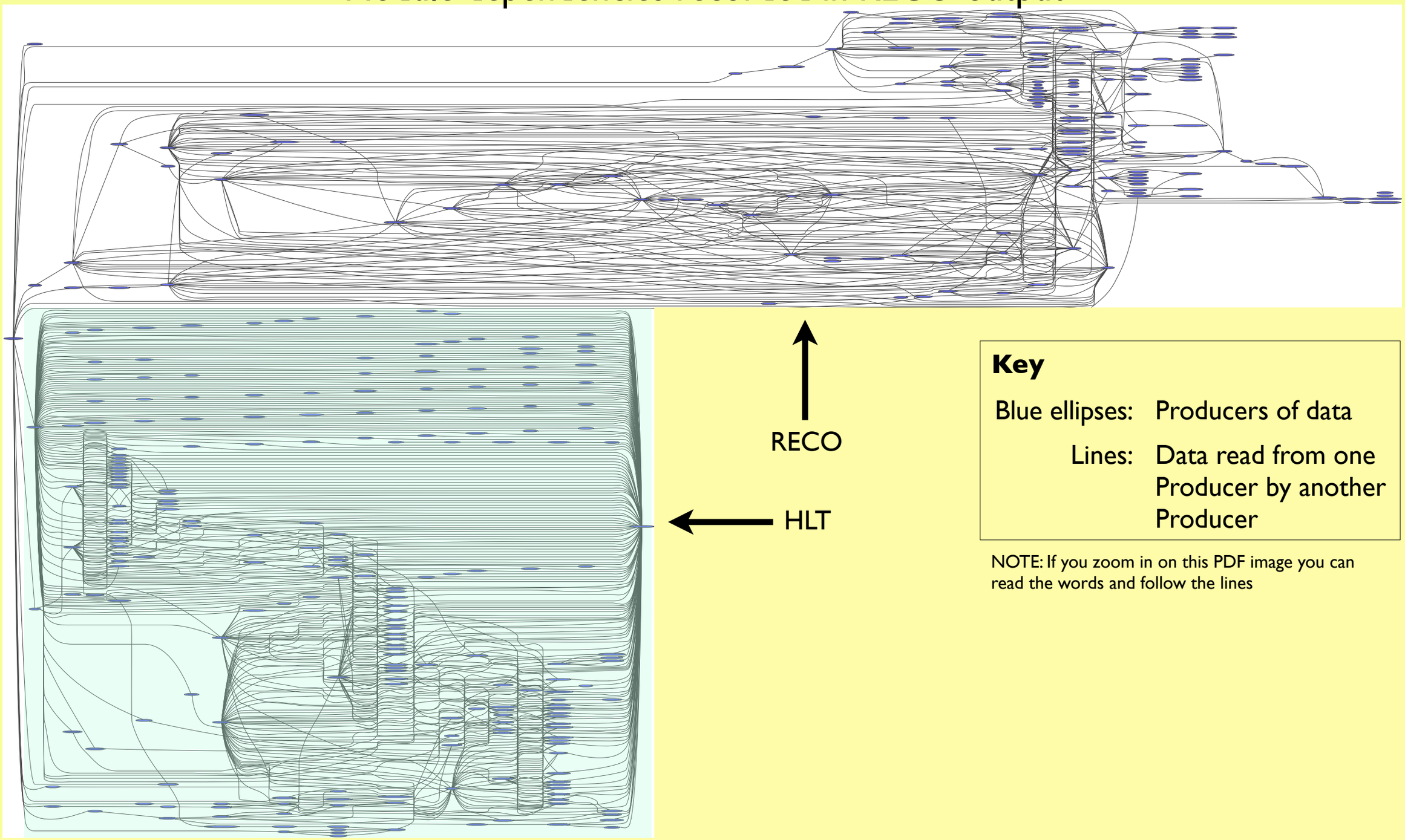


Parentage Provenance

Parentage provenance can be quite large

Recorded event by event

Module dependencies recorded in RECO output



Key

Blue ellipses:	Producers of data
Lines:	Data read from one Producer by another Producer

NOTE: If you zoom in on this PDF image you can read the words and follow the lines

Skimming

Can customize how much provenance to drop

Data origin and storage relation	Drop specification			
	None	**Dropped**	**Prior**	**All**
current kept	keep	keep	keep	drop
current ancestor	keep	keep	keep	drop
current unrelated	drop	drop	drop	drop
prior kept	keep	keep	drop	drop
prior ancestor	keep	drop	drop	drop
prior unrelated	drop	drop	drop	drop

Where

Current: data created in current process

Prior: data created in prior process

Kept: the job configuration stated the data should be saved

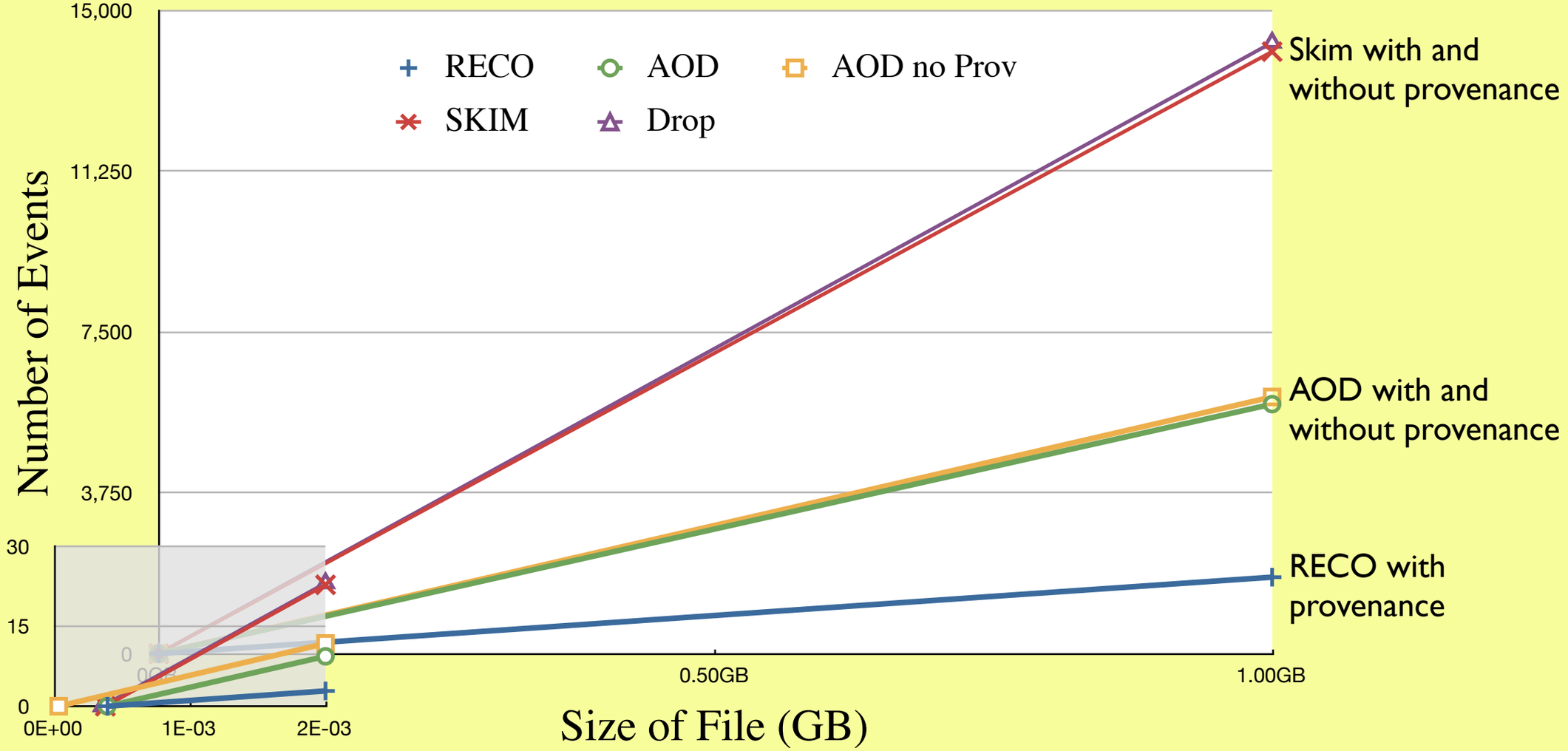
Ancestor: the data was not marked to be saved but the data was used directly or indirectly to create kept data

Unrelated: the data was not marked to be saved and was not used to create kept data

RECO and AOD use drop None while skims may use any

Size Comparison

	RECO	AOD	Skimmed data with drop specification			
			None	**Dropped**	**Prior**	**All**
Data/ev (bytes)	555,386	167,054	70,092	70,092	70,092	70,092
Prov/ev (bytes)	4927	4743	1124	58	2	2
overhead @ 1 GB	0.9%	2.9%	1.6%	0.12%	0.04%	0.04%



Conclusions

CMS's data processing framework records provenance of

Event selections

Data product construction

We have attempted to minimize the disk cost of the provenance

Allow trimming of provenance to meet the physics needs

Have gone through several iteration with at least one more planned

Experts have used detailed information directly from the file

Have used what modules read which data products info for threading studies

We have a first generation tool for inspecting the provenance

Given a file we can easily dump

- the configuration of each of the Producers
- what data products each produced
- what event selection criteria were used

Have had positive feedback from physicists who use the tools

Plan on expanding the tools

- e.g., allow easy access to per event per product provenance