informatics mathematics

Early experience using Apache for a query base scientific visualization Infrastructure.

Bruno Raffin
DataMove

bruno.raffin@inria.fr

September 2016

Warning

We are getting experience with Bid Data, but reported issues may come from miss-configuration as much as internal limitations of the tools

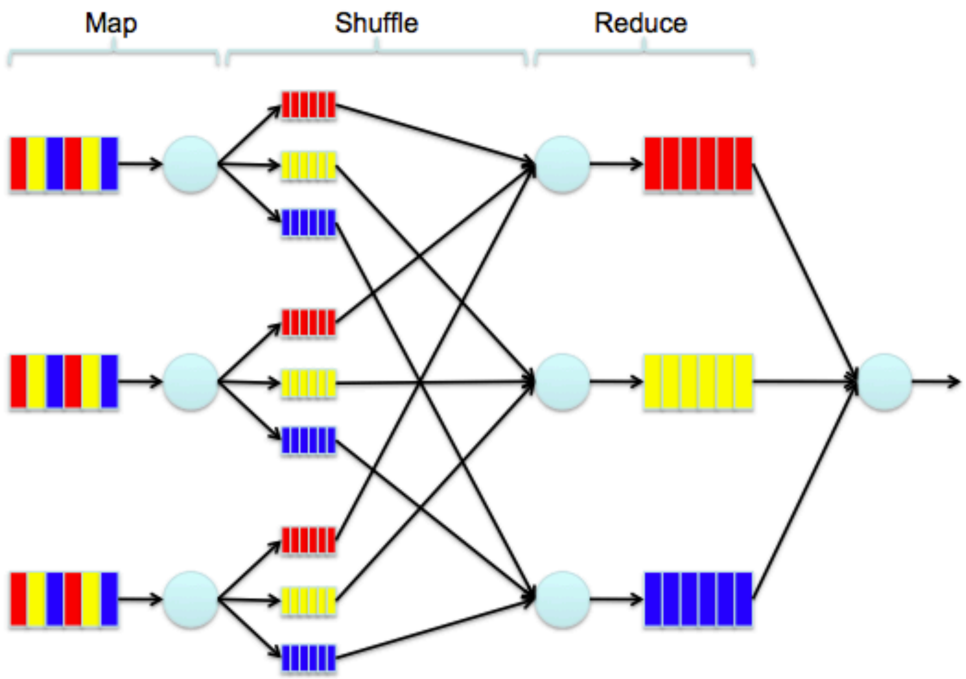
2 part talks:

1. Map/Reduce and Flink overview
2. Early Experience using Apache for scientific data

Big Data: Google Map/Reduce

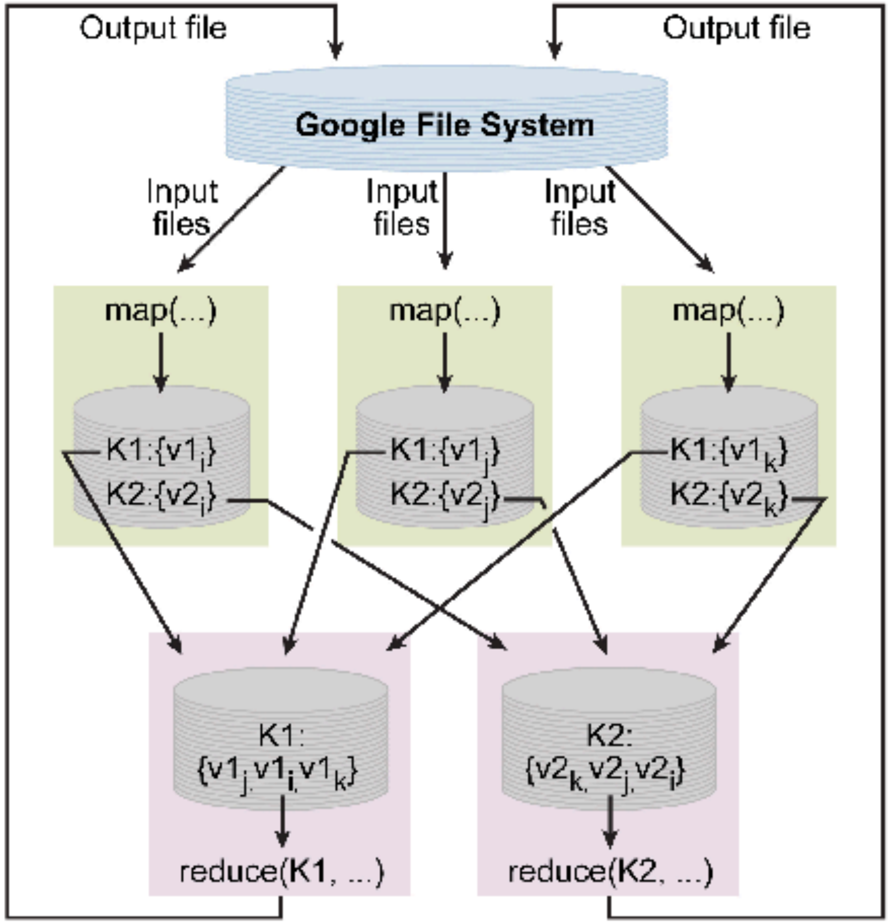
Google Map/Reduce (2004):

- Two data parallel operators: map, reduce
- Values are indexed with a key (key/value model)
- Parallel execution on a cluster (distributed memory)
- Runtime takes care of tasks scheduling, load balancing and fault tolerance



Map/Reduce Programming Model

Map: (K_i,v_i) -> List (K_j,v_j)
Reduce: $(K_q$, list $(v_q))$ -> (K_q, v'_q)



Example: Word Count

Input files
Foo.txt: “Sweet, this is the foo file”
Bar.txt: “This is the bar file”

mapper (filename, file-contents):
  for each word in file-contents:
    emit (word, 1)

reducer (word, values):
  sum = 0
  for each value in values:
    sum = sum + value
  emit (word, sum)

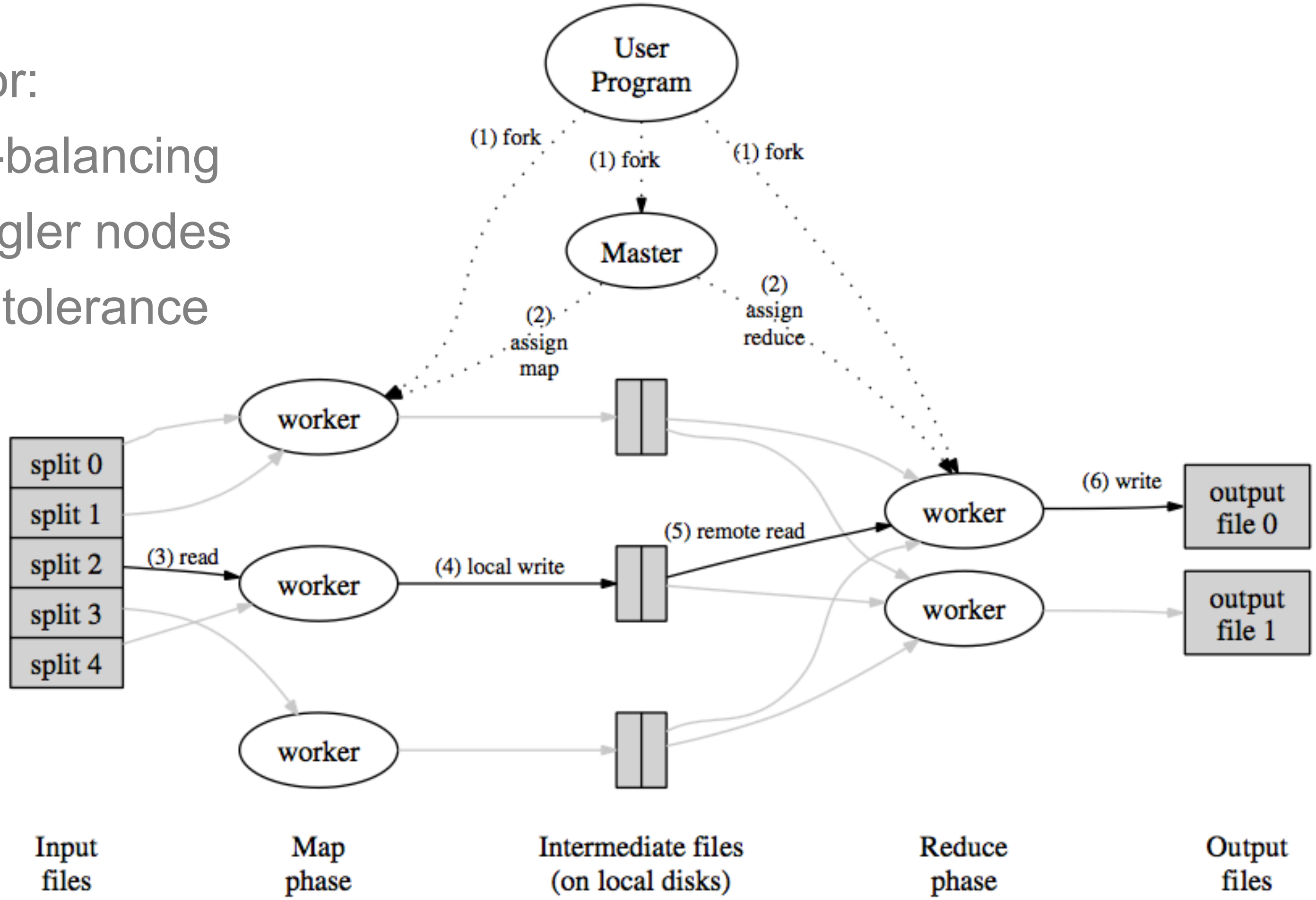
Output

sweet	1
this	2
is	2
the	2
foo	1
bar	1
file	2

Dynamic Task Scheduling

Support for:

- Load-balancing
- Straggler nodes
- Fault tolerance



Original MapReduce

Main open implementation: Hadoop Map/reduce

Limitations:

- Though the map and reduce operation are universal, it is difficult to fit some algorithms (performance-wise and programming-wise)
- Results (included intermediate ones) are written to disk (performance issue)
- Target cloud rather than HPC platforms

In front of these limitations **new frameworks emerged**: Piccolo, Spark, **Flink**, …

The Flink Case

A recent framework, called Stratosphere before to join the Apache Foundation

Performance improvements:

- **Intermediate results are stored in-memory** (unless explicitly stated)
- Intermediate results are mutable (in opposition to Spark RDDs)

The Flink Case: Programmability + Performance

More parallel operators

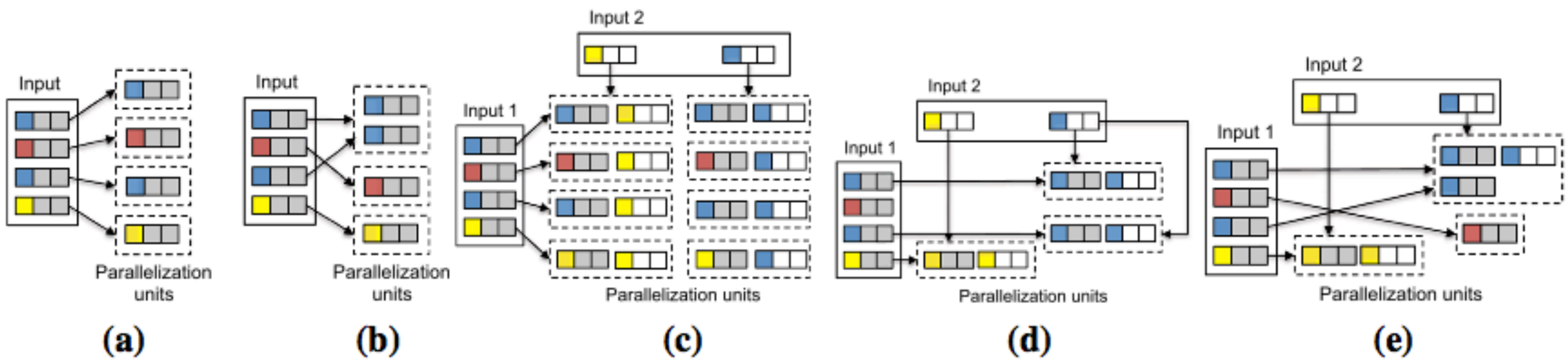


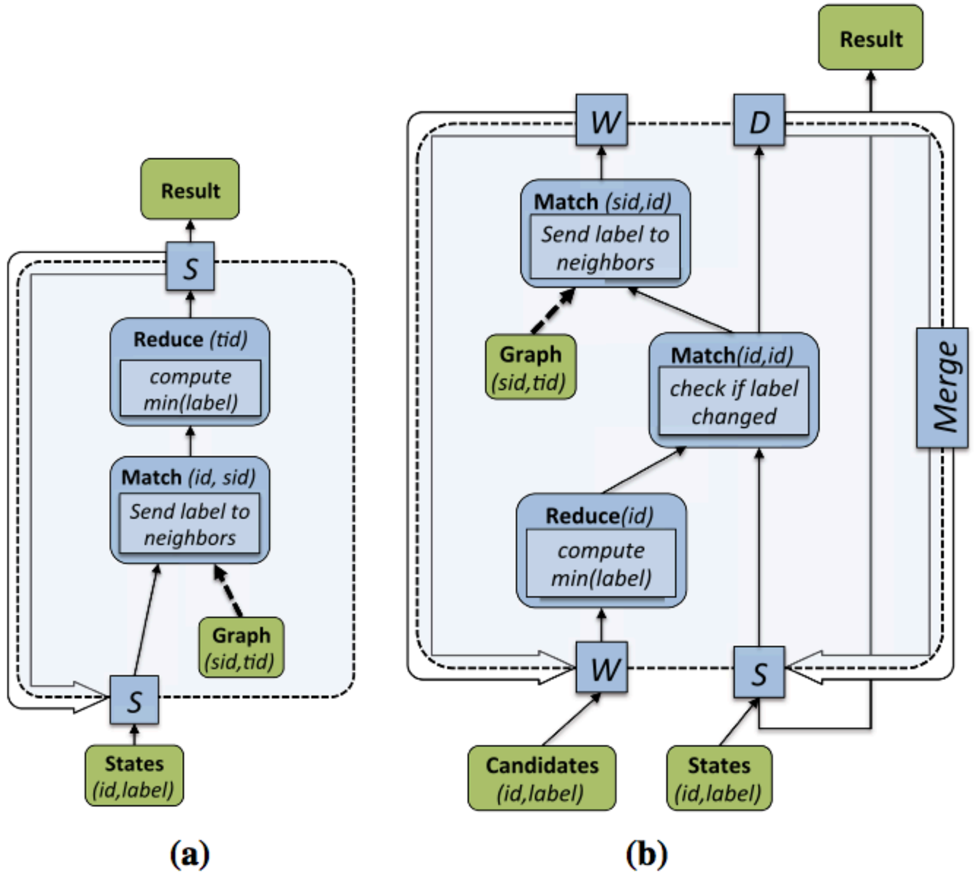
Fig. 5 The five second-order functions (PACTs) currently implemented in Stratosphere. The parallelization units implied by the PACTs are enclosed in *dotted boxes*. **a** Map **b** Reduce **c** Cross **d** Match **e** CoGroup

The Flink Case: Programmability + Performance

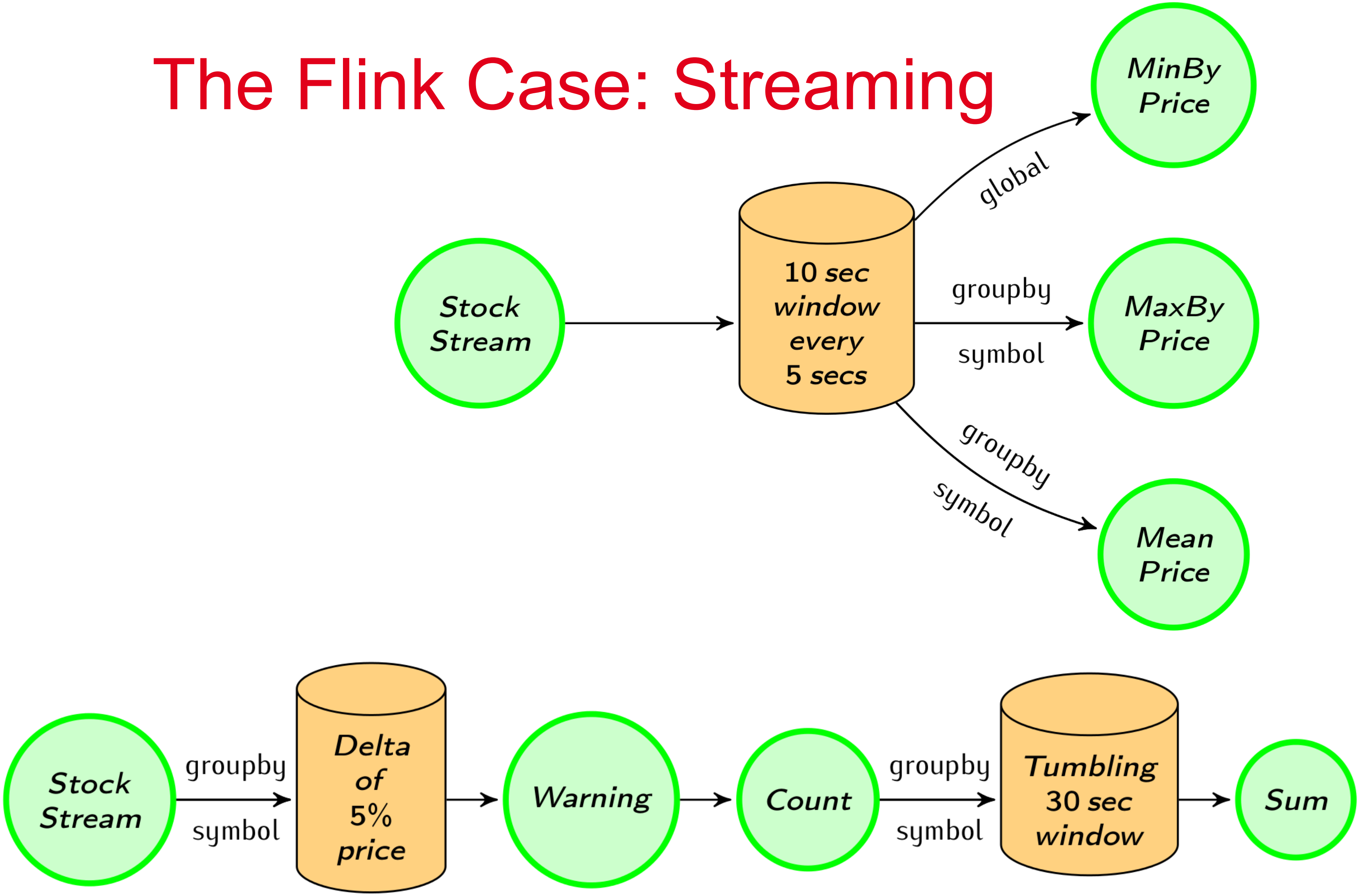
Programmable data flow graph

(support iterative algorithms)

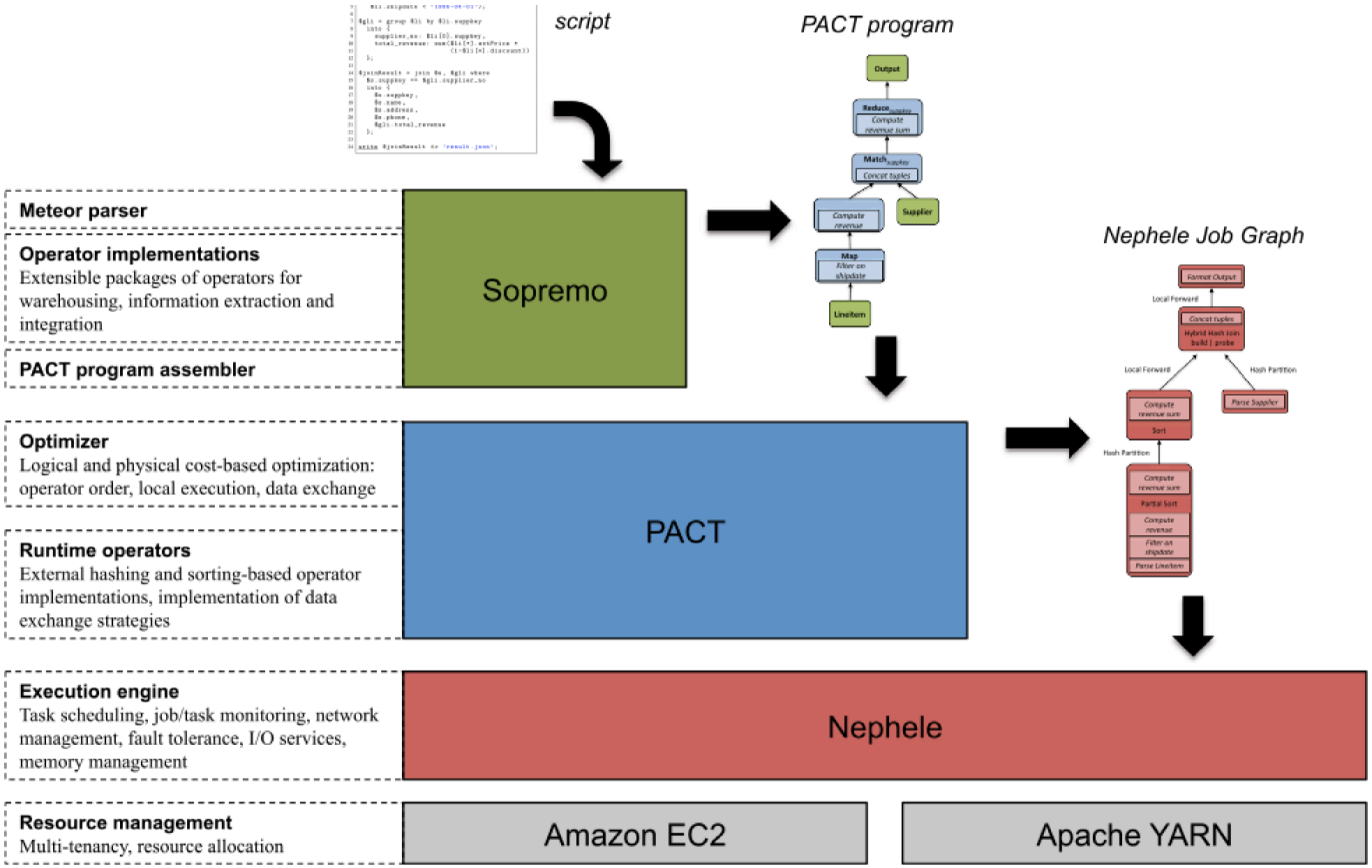
Fig. 6 An algorithm that finds the connected components of a graph as a bulk iteration and an incremental Stratosphere iteration. **a** Bulk iteration **b** Incremental iteration



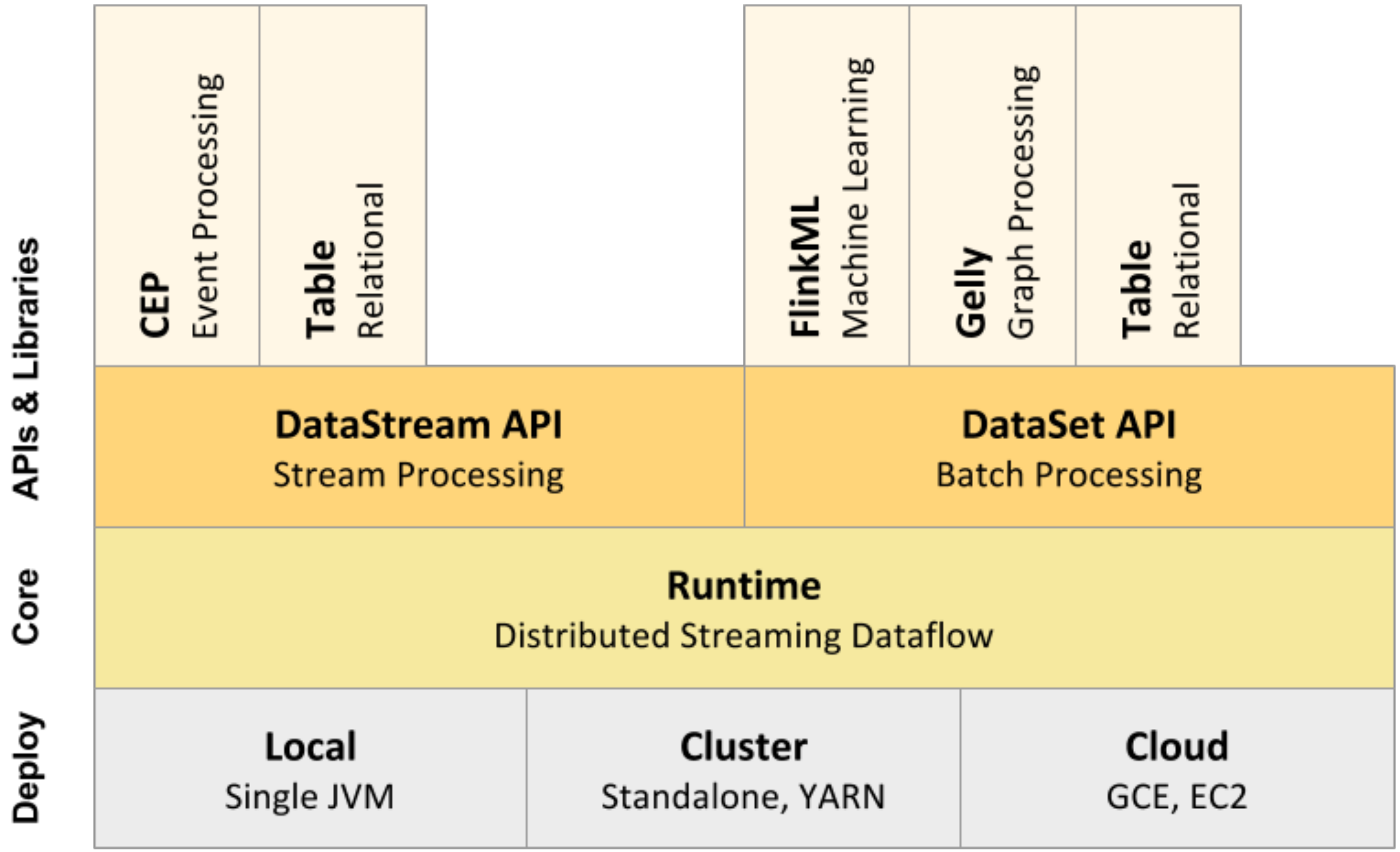
The Flink Case: Streaming



The Flink Case: Compile+Run Time Optimizations



The Flink Case: Speciallization Layers



File System and Databases

A key component of Big Data frameworks

Base concept: relational databases do not scale, go for key/value oriented storage

The Hadoop classics: HDFS (file system), H-base (column-oriented database)

- Write once, read many times
- Manage data replication for fault tolerance

HPC and Big Data

Running Big Data frameworks on HPC architectures

1. Adapt existing frameworks:
    - RDMA-based Spark http://hibd.cse.ohio-state.edu/
    - Support for Luster (instead of HDFS)
2. Develop new frameworks (MPI based):
    - Picollo
    - MapReduce- MPI

Using Big Data frameworks to analyse HPC data (simulation results, traces,…)

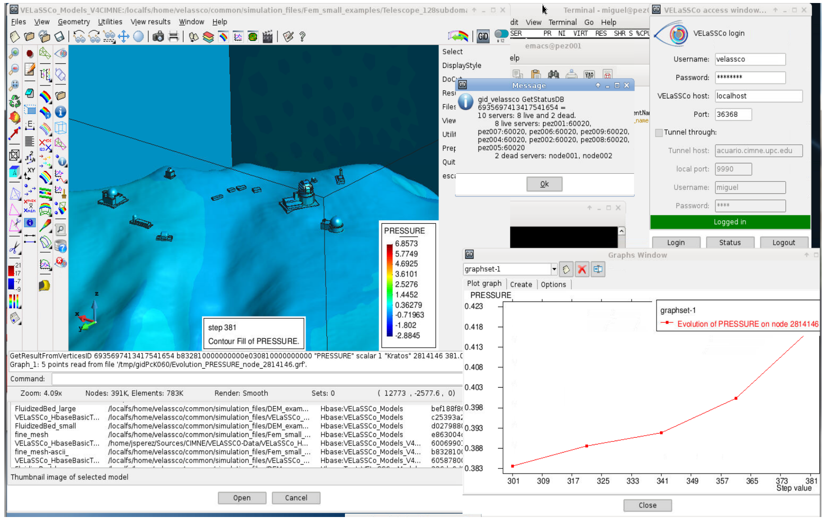
VelaSSco (FP7)

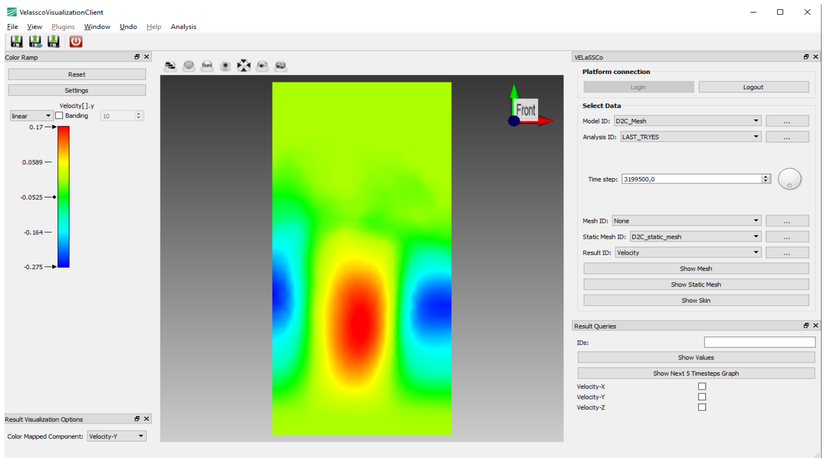
Query based Scientific Visualization:

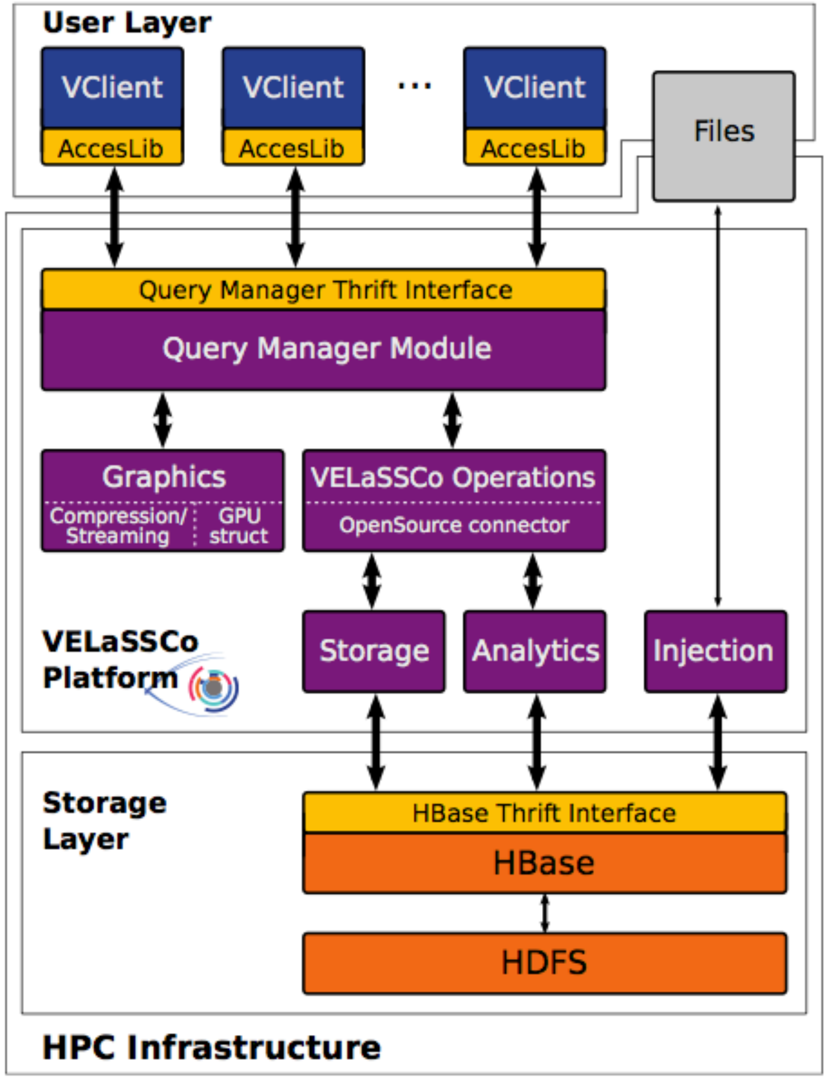
- FEM/DEM simulation data
- Hadoop software suite (MapReduce, HDFS, Hbase, Yarn, Thrift)
- Key/value: (timestep+rank-id, data)
- Scientist request some visualization (isosurface for a given timestep):

Vis client <-> front server <-> map/reduce job <-> HBASE

Velassco Infrastructure







HBase

Column-oriented database

COLUMN FAMILIES

Row key	personal data		professional data	
empid	name	city	designation	salary
1	raju	hyderabad	manager	50,000
2	ravi	chennai	sr.engineer	30,000
3	rajesh	delhi	jr.engineer	25,000

Internally rely on key/value storage: one key per column

Rows are sorted according to key

Data are splits in blocks (10GB first changed to 256MB) , triplicated and distributed on the cluster disks.

HBase

Data are splits in blocks (10GB first changed to 256MB) , triplicated and distributed on the cluster disks.

At most one mapper per :

too few splits will impair parallelization

Hbase support virtual splits (virtually “split” a split in X parts to enable X mappers to work concurrently) – We tried and experienced issues for X>=8

A mapper can access a split locally or remotely (slower)

HBase issues

1. Initial perf. limitation: not enough splits -> not enough parallelism
2. Data sharding: be carefull to hot spot

Slow changing bit Fast changing bit

First key: (model-id,analysis-id,**timestep,rank-id**)

New key: (model-id,**rank-id**,analysis-id,**timestep**)

Rank-id: MPI partition adopted by the numerical simulation that produced the data set.

Queries are often working on a single timestep. *First key* tends to have all data from a timestep in the same split -> low parallelism. *New key* lead to a better spread of different ranks data. But….

HBase Issues

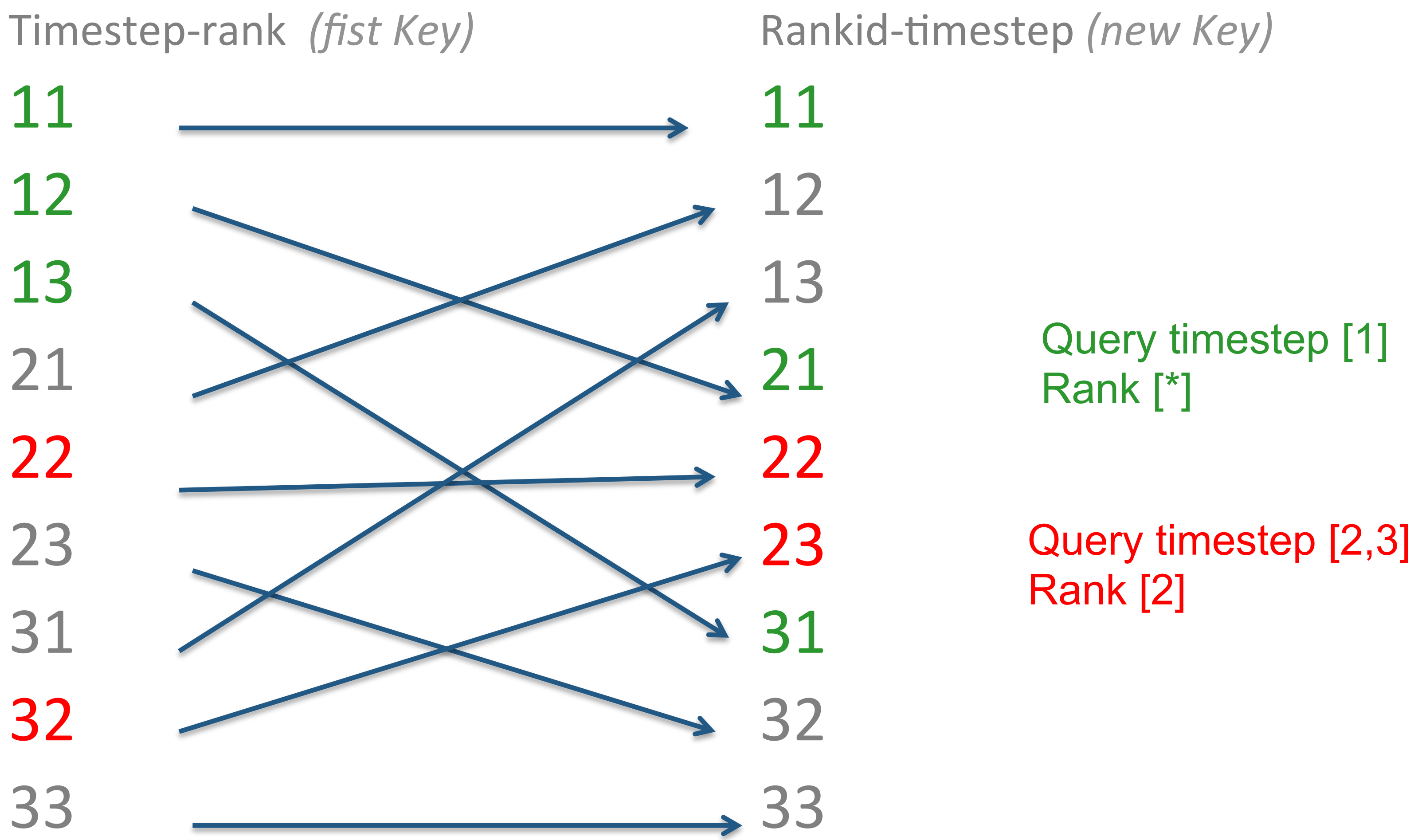
But it depends on the query and data:

DEM (large number of particles): many splits (1000s) so even first key lead to good results

FEM (mesh): smaller number of splits (100s)

Discret-to-Continuous query requires to have a window of time steps, but is only applied to a given number of partitions: *first Key* works better because….

HBase Issues



Contiguous data: more likely to be in same split

Data Injection into HBase

Data inflation: 5x (3x for triplication remaining for metadata from column split ?)

Time to inject: 0.5TB -> 2 days

From the simulation to Hbase: use Flume to grab files produced by the simulation (one file per timestep and rank) and inject them into Hbase.

Cluster: A 10 nodes dedicated cluster – 110 TO storage – 10 GB Ethernet network

Permanent Apache installation. Multi-user. Yarn in charge of job scheduling.

Map/reduce versus Spark versus Flink

From map/reduce to Flink

- Query translation: almost direct (Flink -> tuples, Map/reduce key/values)
- HBASE connector in Flink is immature
- Performance: GetBoundaryofaMesh (mesh surface)
  - Map/reduce 22s, Flink 12s
  - Early tests with Spark: 15s ? (no trouble with the HBASE connector)

Summary

- Apache map/reduce: mature but store intermediate results to disks
- Spark /& Flink: in memory storage of intermediate results.
- Flink: promising (known in particular for its streaming capabilities) but for the moment code less mature than Spark.

- HDFS/Hbase: not very satisfied
- Try with Cassendra?? ("everyone knows that HDFS/Hase is portable but slow" ?)

Are Big Data tools suited for scientific data?

HPC versus Big Data

HPC

- Numerical simulations
- Thin software stack
- Supercomputer
- C/C++/Fortran/Python
- Looking for the universal programming model
- Small Market

Big Data

- Web and business data
- Thick software stack
- Cloud
- Java/Scala
- Many domain specific languages (DSL)
- Large Market

US National Strategic Computing Initiative (2015)

Sec. 2. Objectives. Executive departments, agencies, and offices (agencies) participating in the NSCI shall pursue five strategic objectives:

1. Accelerating delivery of a capable exascale computing system that integrates hardware and software capability to deliver approximately 100 times the performance of current 10 petaflop systems across a range of applications representing government needs.
2. **Increasing coherence between the technology base used for modeling and simulation and that used for data analytic computing.**

…..

FIN