Big Data Management and Scalable Data Science: Challenges and (some) Solutions

Prof. Dr. Volker Markl
http://www.user.tu-berlin.de/marklv/
http://www.dima.tu-berlin.de
http://www.dfki.de/web/forschung/iam
http://bbdc.berlin

volker.markl@tu-berlin.de

Data & Analysis: Increasingly Complex!

scalability

DM

data volume too large	***V**olume*	Reporting	aggregation, selection
data rate too fast	***V**elocity*	Ad-Hoc Queries	SQL, XQuery
data too heterogeneous	***V**ariability*	ETL/ELT	MapReduce
data too uncertain	***V**eracity*	Data Mining	MATLAB, R, Python
		Predictive/Prescriptive	MATLAB, R, Python

ML

algorithms

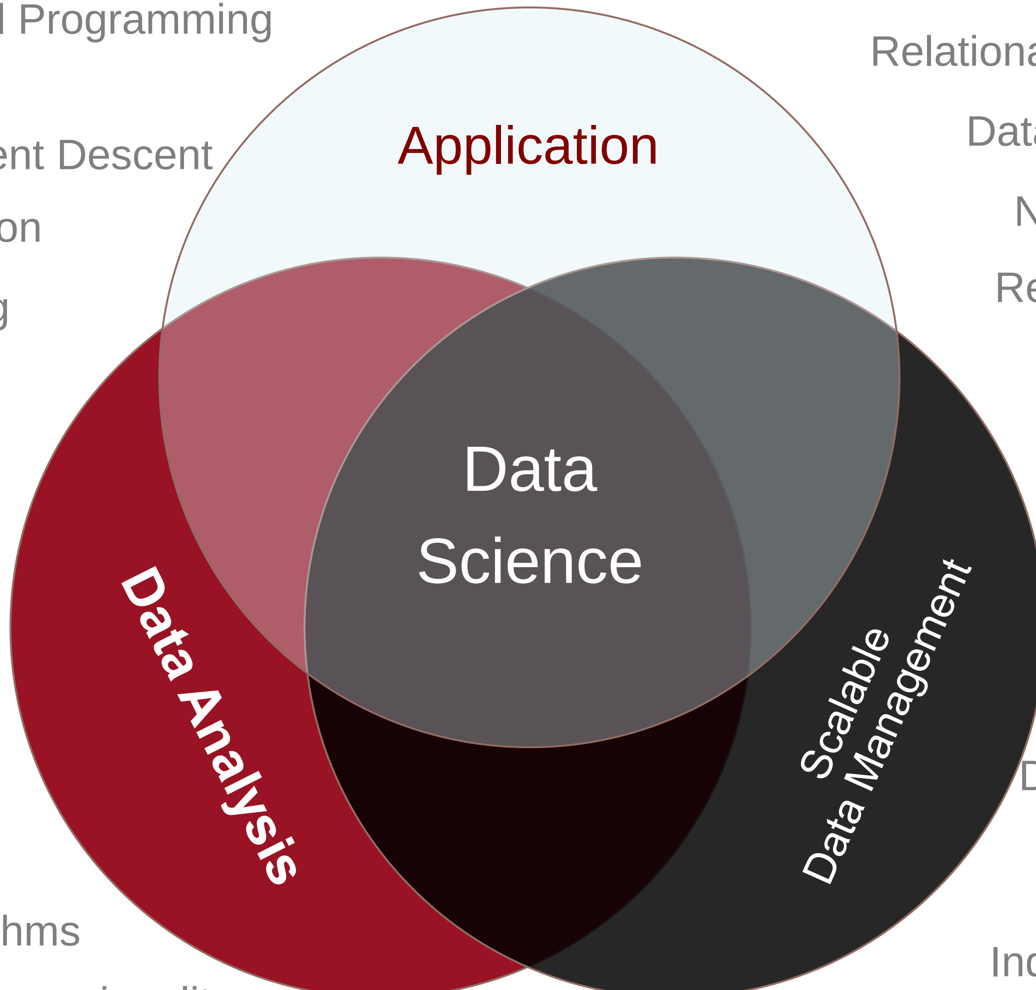
Data

Analysis

"Data Scientist" – "Jack of All Trades!"

Domain Expertise (e.g., Industry 4.0, Medicine, Physics, Engineering, Energy, Logistics)

Mathematical Programming
Linear Algebra
Stochastic Gradient Descent
Error Estimation
Active Sampling
Regression
Monte Carlo
Statistics
Sketches
Hashing
Convergence
Decoupling
Iterative Algorithms
Curse of Dimensionality

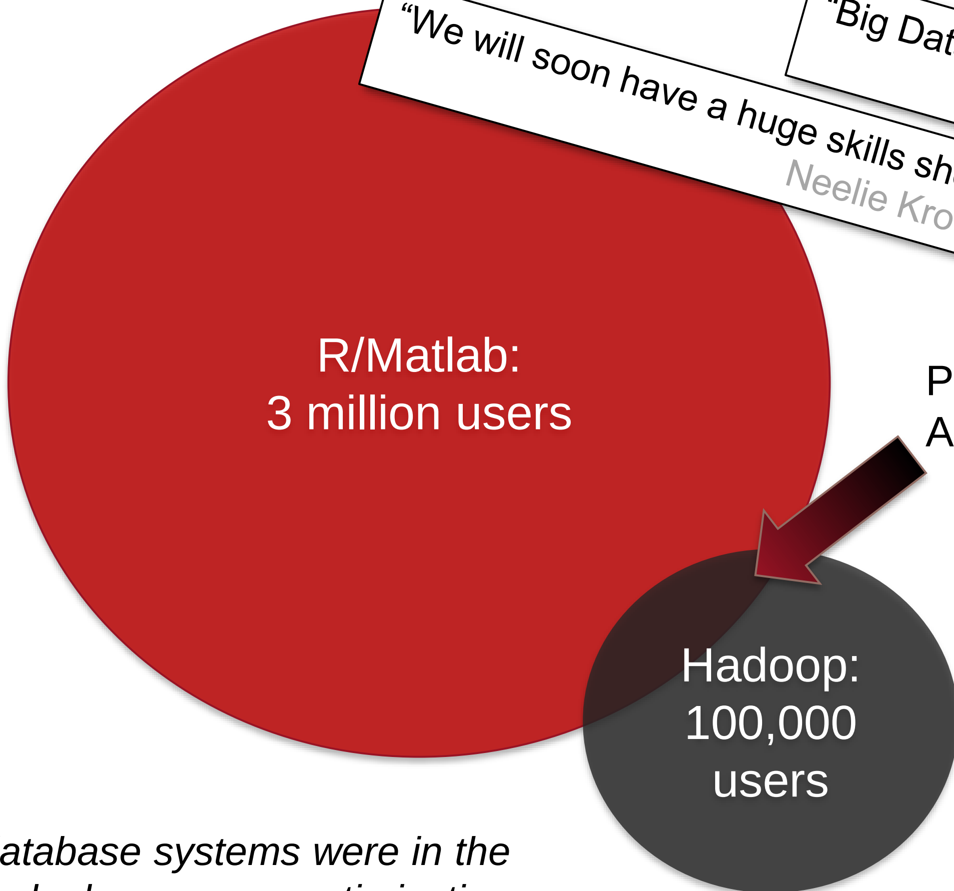


Relational Algebra / SQL
Data Warehouse/OLAP
NF^2 /XQuery
Resource Management
Hardware Adaptation
Fault Tolerance
Memory Management
Parallelization
Scalability
Memory Hierarchy
Data Analysis Language Compiler
Query Optimization
Indexing
Data Flow
Control Flow
Real-Time

New Technology to the Rescue!

Big Data Analytics Requires Systems Programming

Data Analysis
Statistics
Algebra
Optimization
Machine Learning
NLP
Signal Processing
Image Analysis
Audio-,Video Analysis
Information Integration
Information Extraction
Data Value Chain
Data Analysis Process
Predictive Analytics



"We will soon have a huge skills shortage for data-related jobs."
Neelie Kroes (ICT 2013, Nov. 7, Vilnius)

"Big Data's Big Problem: Little Talent"
Wall Street Journal

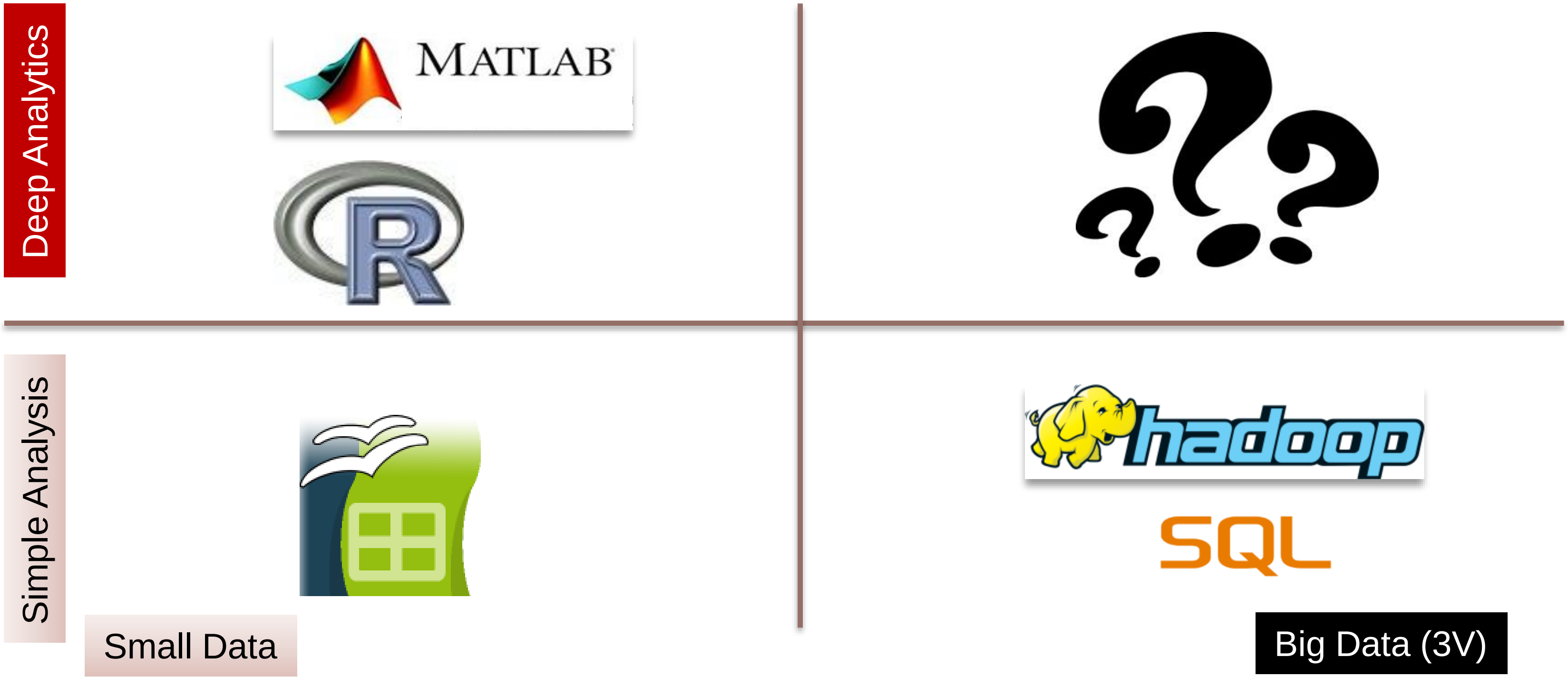
People with Big Data Analytics Skills

Indexing
Parallelization
Communication
Memory Management
Query Optimization
Efficient Algorithms
Resource Management
Fault Tolerance
Numerical Stability

Big Data is now where database systems were in the 70s (prior to relational algebra, query optimization and a SQL-standard)!

Declarative languages to the rescue!

Deep Analysis of „Big Data“ is Key!



Many new companies and products are emerging to enable deep big data analysis; **strong European contenders** include Apache Flink, Parstream, and Exasol. **„New companies“ are the (b)leading users of these technologies,** e.g., in the information economy (e.g., Zalando, Amazon, Researchgate, Soundcloud, Spotify). **„Traditional Big companies“ are following** and still determining strategies (Industrie 4.0, Logistics, Telco, etc.). Most **SMEs are not ready yet to capitalize on Big Data**.

Challenge: Technologies for Data Science at the Intersection of Data Management and Machine Learning

DM

Declarative Languages
Automatic Adaption
Scalable processing

Think ML-algorithms in a scalable way

declarative

Process (iterative) algorithms in a scalable way

ML

Feature Engineering
Representation
Algorithms (SVM, EM, etc.)

aggregation → relational algebra → UDF → iteration/recursion → linear algebra, graph algebrae, etc.

StratoSphere
Above the Clouds

Apache Flink–
a success story originating in Berlin

Dominic Battré, Stephan Ewen, Fabian Hueske, Odej Kao, Volker Markl, Daniel Warneke: Nephele/PACTs: a programming model and execution framework for web-scale analytical processing. SoCC 2010: 119-130

Alexander Alexandrov, Rico Bergmann, Stephan Ewen, et al:
The Stratosphere platform for big data analytics. VLDB J. 23(6): 939-964 (2014)

Paris Carbone, Asterios Katsifodimos, Stephan Ewen, Volker Markl, et al : Apache Flink™: Stream and Batch Processing in a Single Engine. IEEE Data Eng. Bull. 38(4): 28-38 (2015)

Stratosphere: General Purpose Programming + Database Execution

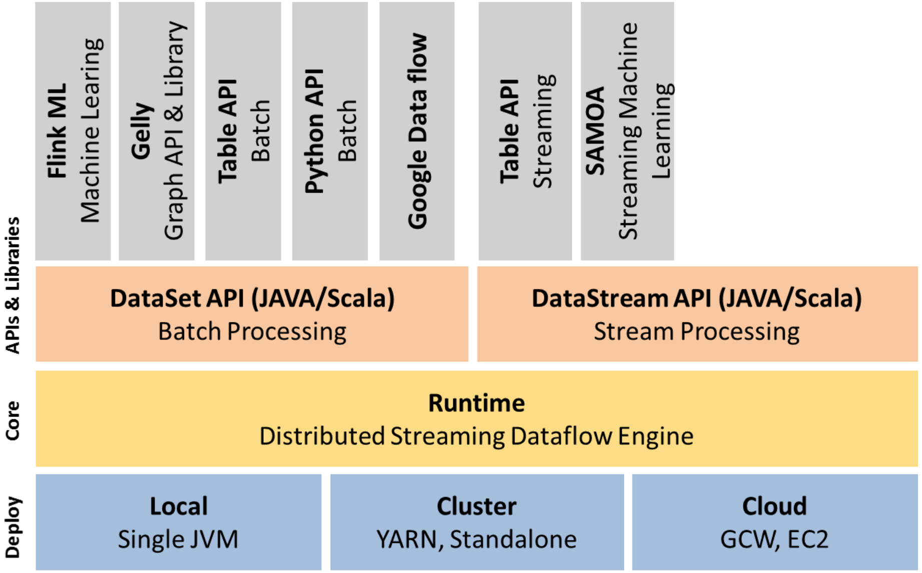
Draws on Database Technology	Adds	Draws on MapReduce Technology
• Relational Algebra • Declarativity • Query Optimization • Robust Out-of-core	• Iterations • Advanced Dataflows • General APIs • Native Streaming	• Scalability • User-defined Functions • Complex Data Types • Schema on Read

8

What is Apache Flink?

Apache Flink is an open source platform for scalable batch and stream data processing.

- The core of Flink is a distributed streaming dataflow engine.
  - Executing dataflows in parallel on clusters
  - Providing a reliable foundation for various workloads
- **DataSet** and **DataStream** programming abstractions are the foundation for user programs and higher layers



http://flink.apache.org

Technology inside Flink

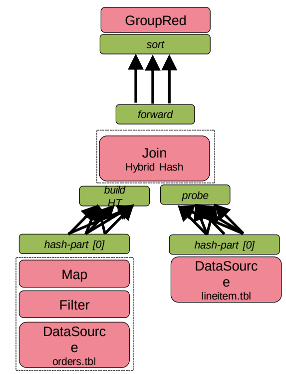
```
case class Path (from: Long, to:
Long)
val tc = edges.iterate(10) {
  paths: DataSet[Path] =>
    val next = paths
      .join(edges)
      .where("to")
      .equalTo("from") {
        (path, edge) =>
          Path(path.from, edge.to)
      }
      .union(paths)
      .distinct()
    next
  }
```

Program

Type extraction stack

Cost-based optimizer

Pre-flight (Client)



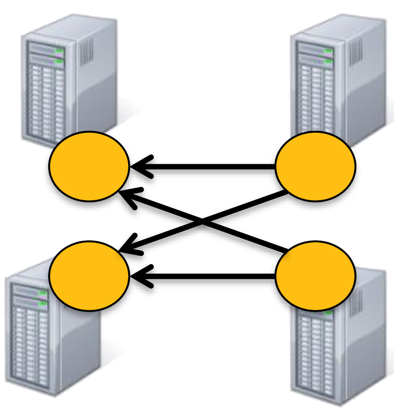
Dataflow Graph

Memory manager

Out-of-core algos

Batch & Streaming

State & Checkpoints



Workers

deploy operators

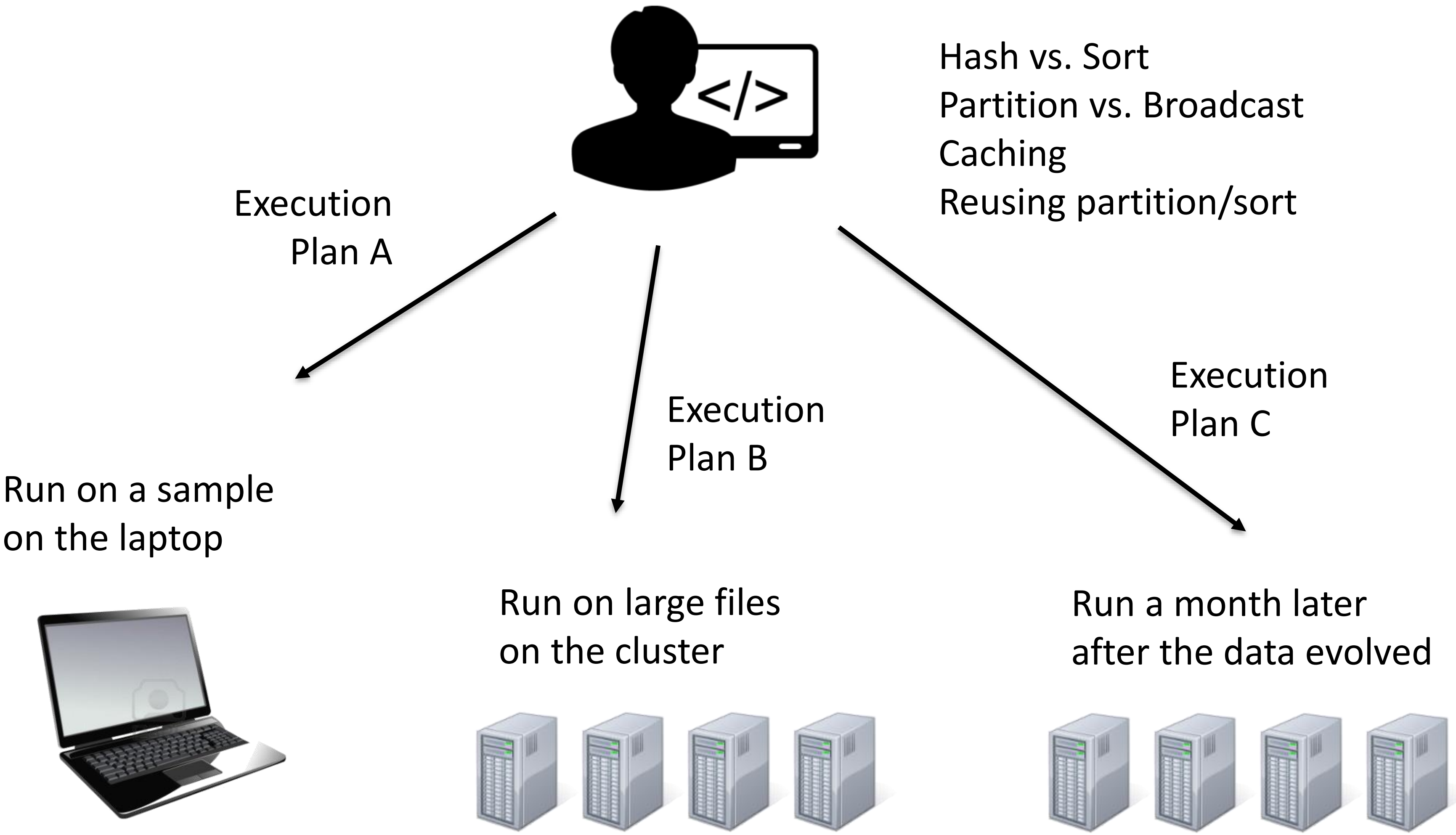
track intermediate results

Recovery metadata

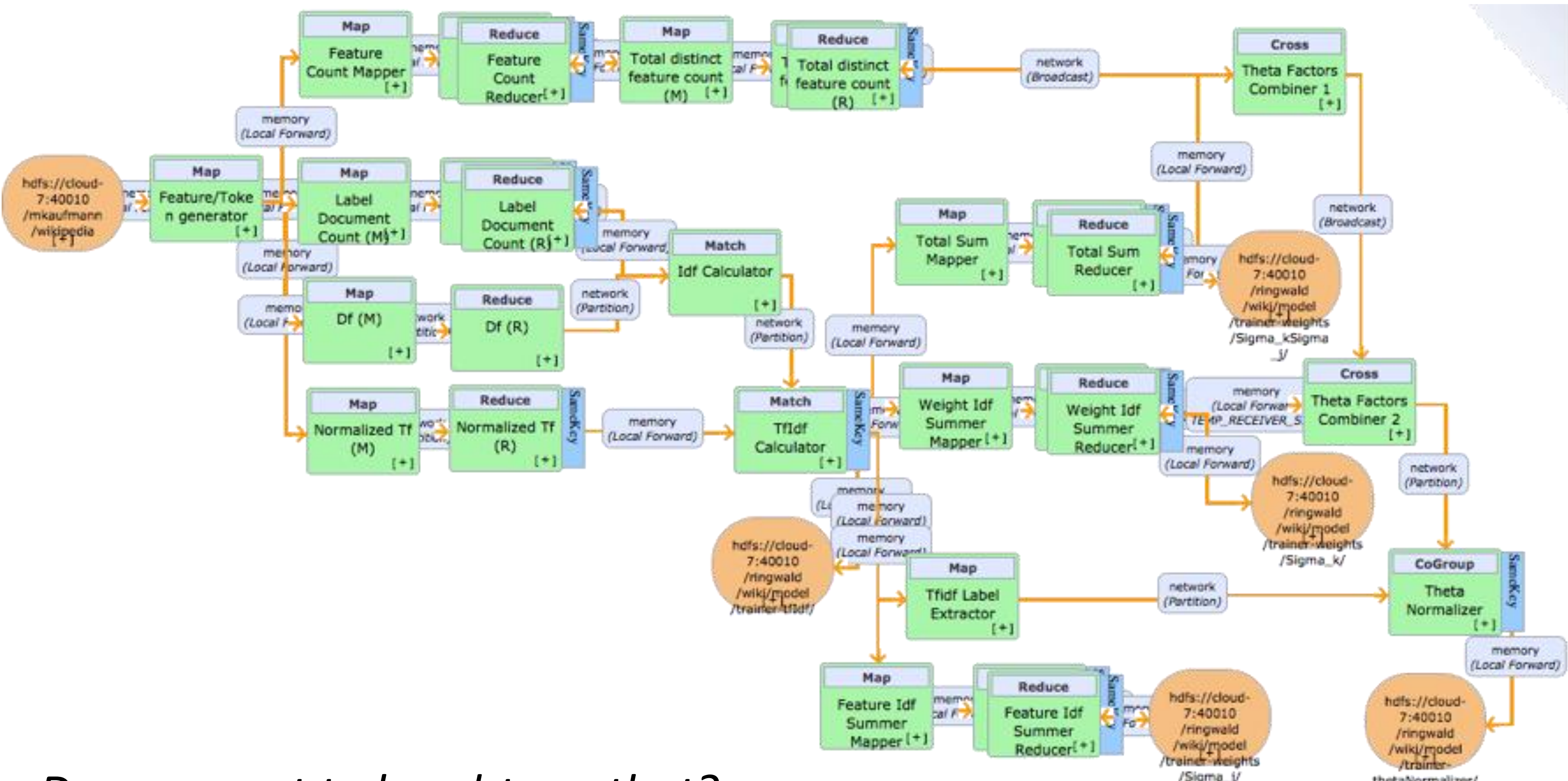
Task scheduling

Master

Effect of optimization



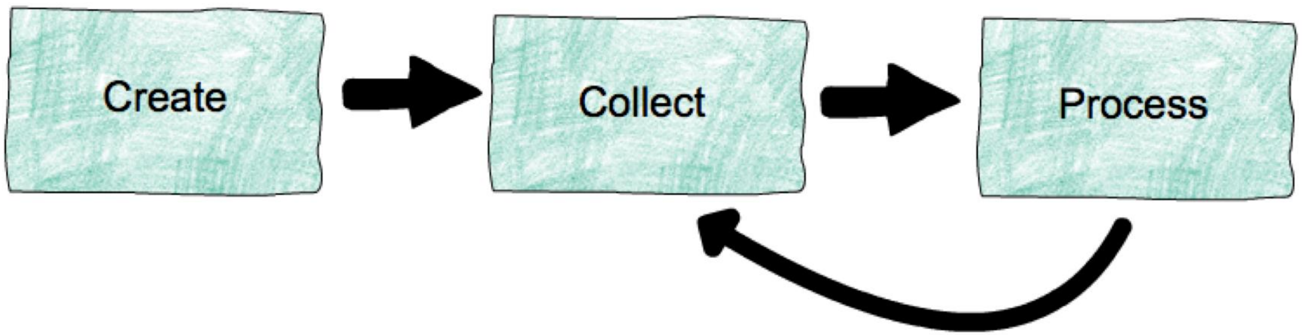
Why optimization ?



Do you want to hand-tune that?

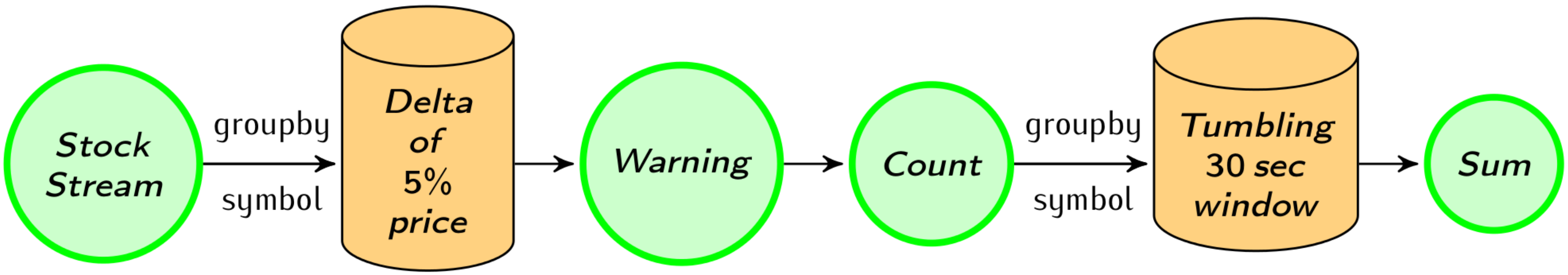
DATA STREAMING ANALYSIS

Life of data streams



- **Create:** create streams from event sources (machines, databases, logs, sensors, …)
- **Collect:** collect and make streams available for consumption (e.g., Apache Kafka)
- **Process:** process streams, possibly generating derived streams (e.g., Apache Flink)

Stream Analysis in Flink




```scala
case class Count(symbol: String, count: Int)
val defaultPrice = StockPrice("", 1000)

//Use delta policy to create price change warnings
val priceWarnings = stockStream.groupBy("symbol")
  .window(Delta.of(0.05, priceChange, defaultPrice))
  .mapWindow(sendWarning _)

//Count the number of warnings every half a minute
val warningsPerStock = priceWarnings.map(Count(_, 1))
  .groupBy("symbol")
  .window(Time.of(30, SECONDS))
  .sum("count")
```

More at: http://flink.apache.org/news/2015/02/09/streaming-example.html

Defining windows in Flink

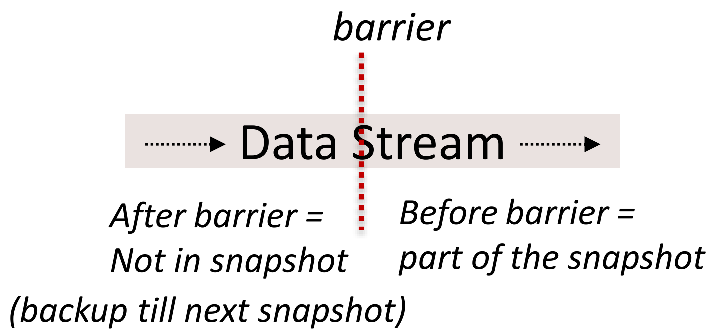
- Trigger policy
  - When to trigger the computation on current window
- Eviction policy
  - When data points should leave the window
  - Defines window width/size
- E.g., count-based policy
  - evict when #elements > n
  - start a new window every n-th element
- Built-in: Count, Time, Delta policies

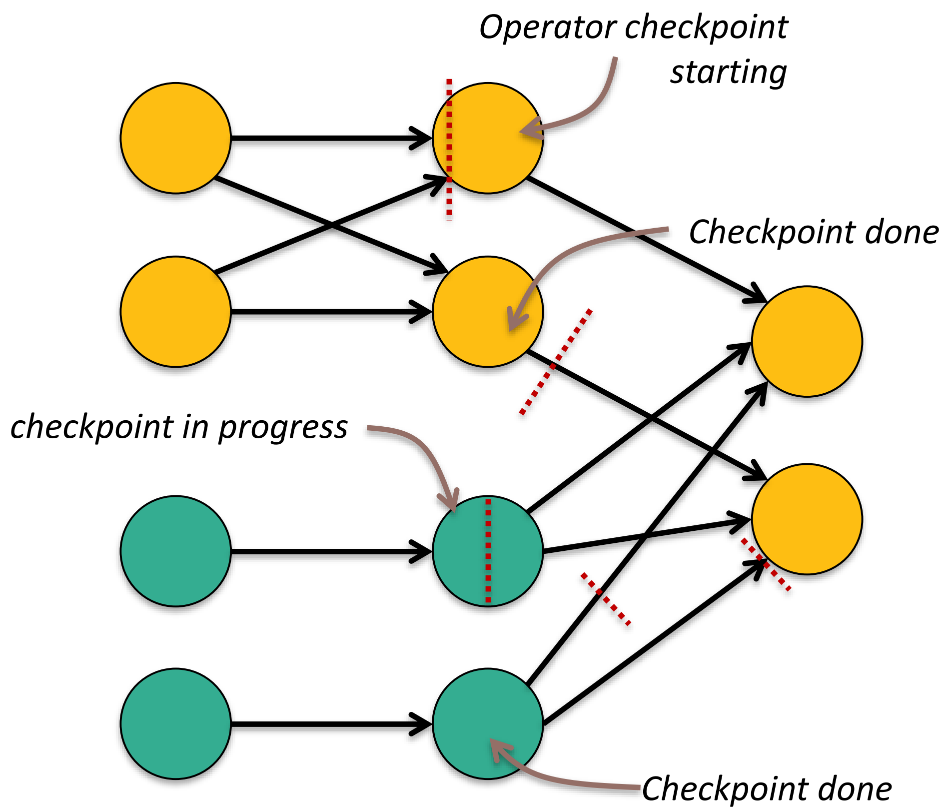
Checkpointing / Recovery

- Flink acknowledges batches of records
  - Less overhead in failure-free case
  - Currently tied to fault tolerant data sources (e.g., Kafka)
- Flink operators can keep state
  - State is checkpointed
  - Checkpointing and record acks go together
- Exactly one semantics for state

Checkpointing / Recovery

Pushes checkpoint barriers through the data flow



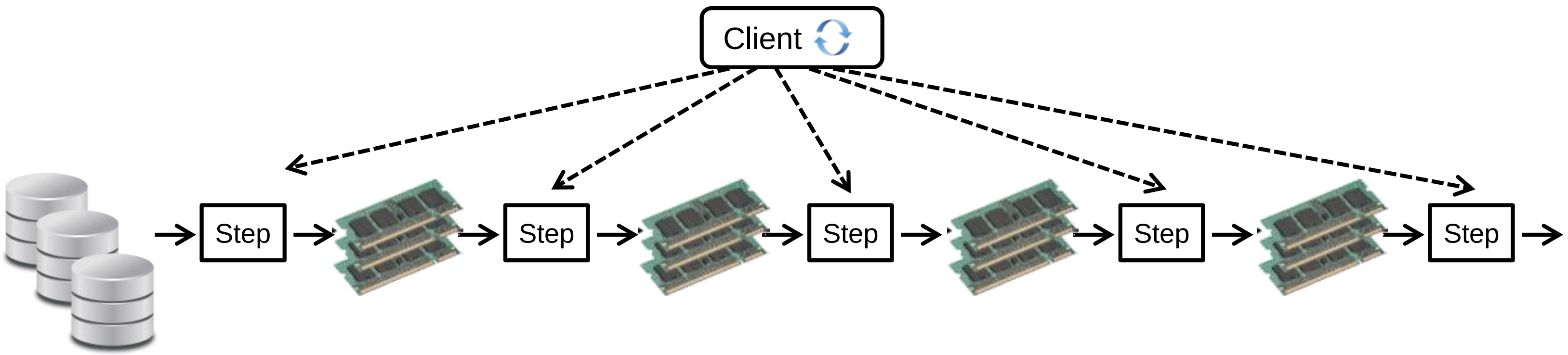


Chandy-Lamport Algorithm for consistent asynchronous distributed snapshots

ITERATIONS IN DATA FLOWS → MACHINE LEARNING ALGORITHMS

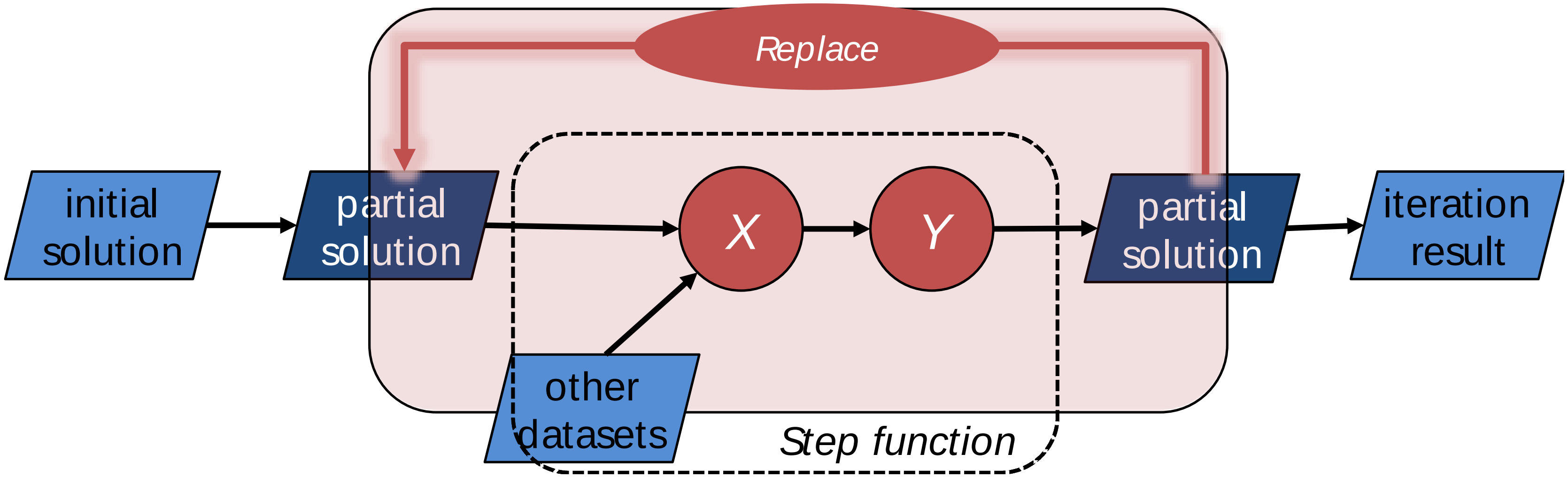
Stephan Ewen, Kostas Tzoumas, Moritz Kaufmann, Volker Markl:
Spinning Fast Iterative Data Flows. PVLDB 5(11): 1268-1279 (2012)

Iterate by looping



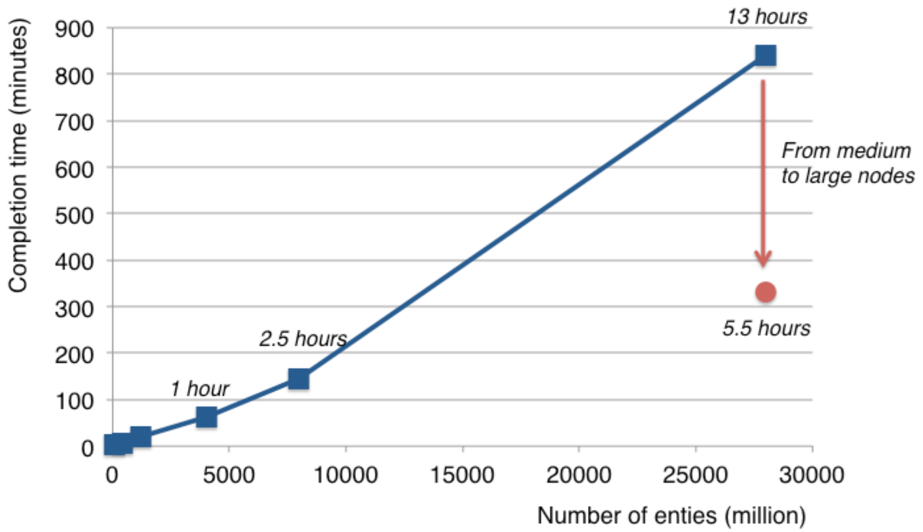
- for/while loop in client submits one job per iteration step
- Data reuse by caching in memory and/or disk

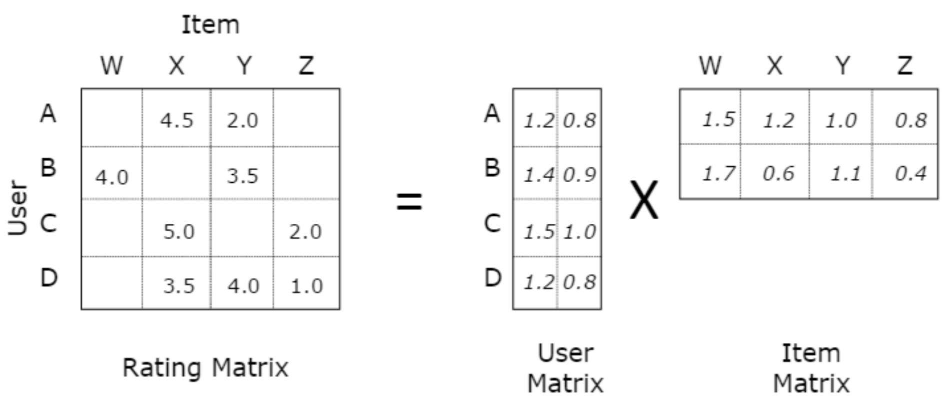
Iterate in the Dataflow



Large-Scale Machine Learning

Factorizing a matrix with 28 billion ratings for recommendations





(Scale of Netflix or Spotify)

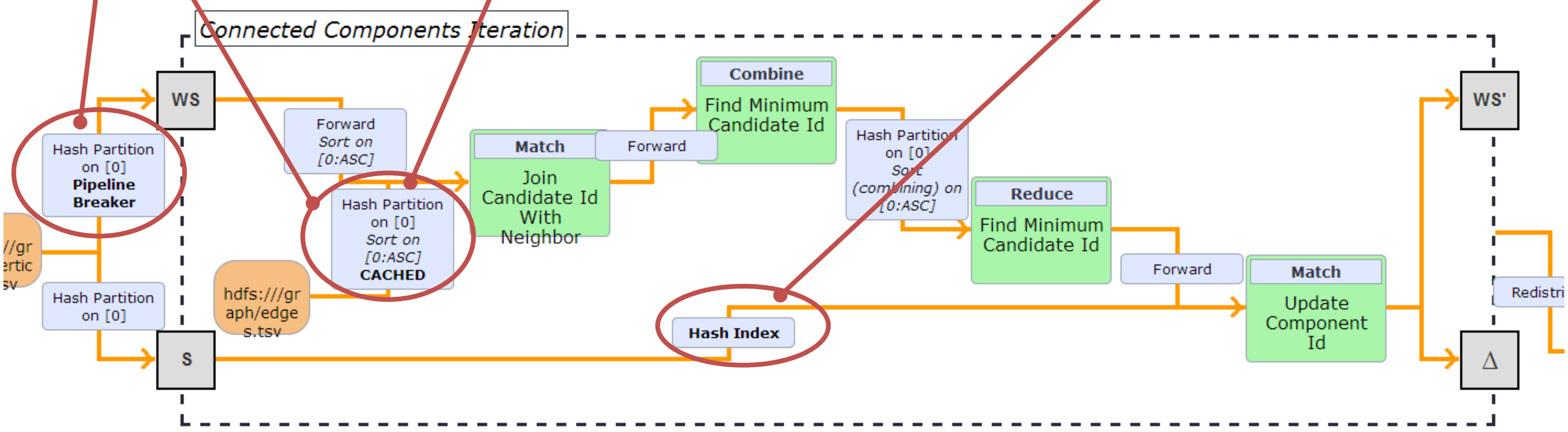
More at: http://data-artisans.com/computing-recommendations-with-flink.html

Optimizing iterative programs

Pushing work „out of the loop“

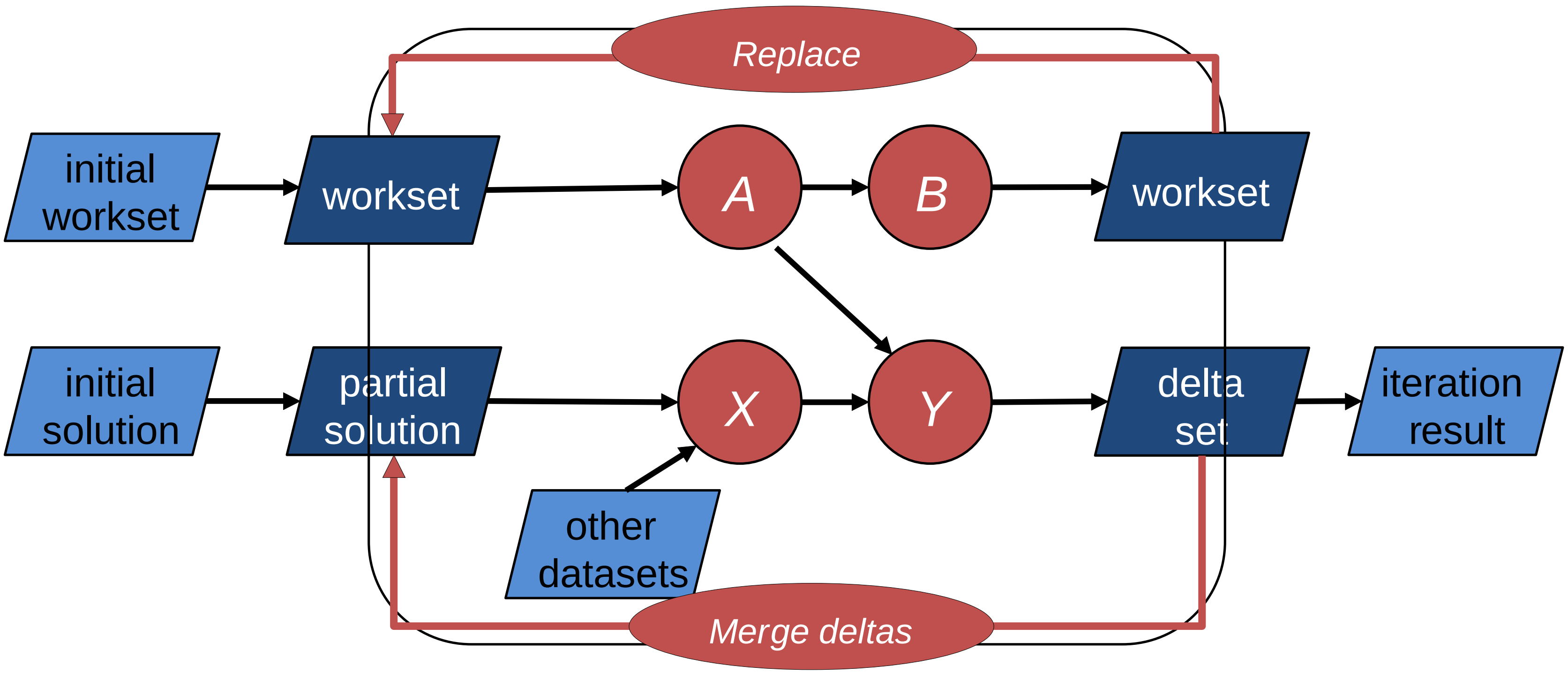
Caching Loop-invariant Data

Maintain state as index

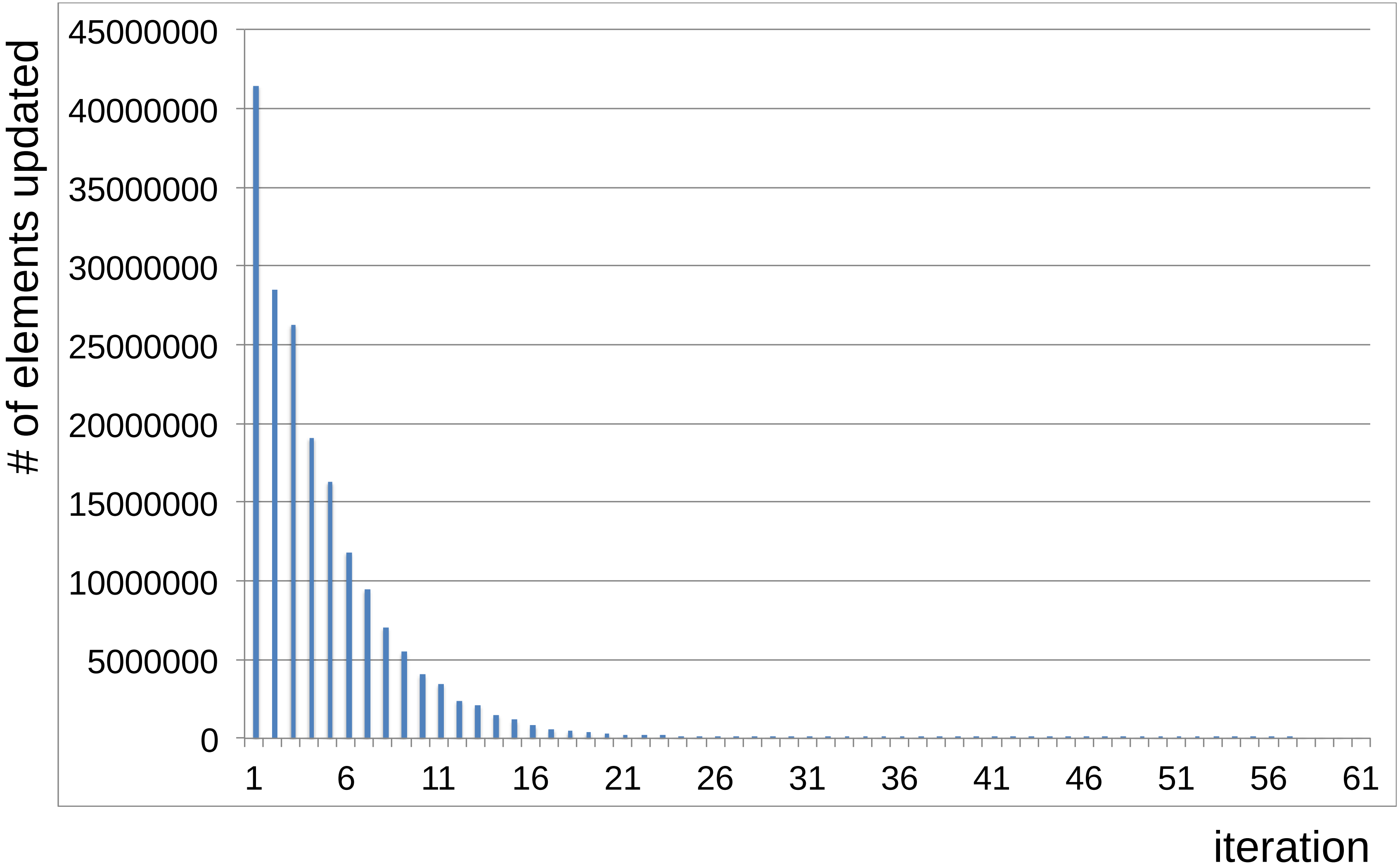


STATE IN ITERATIONS
→ GRAPHS AND MACHINE LEARNING

Iterate natively with deltas

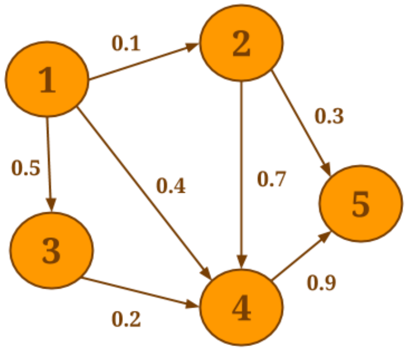


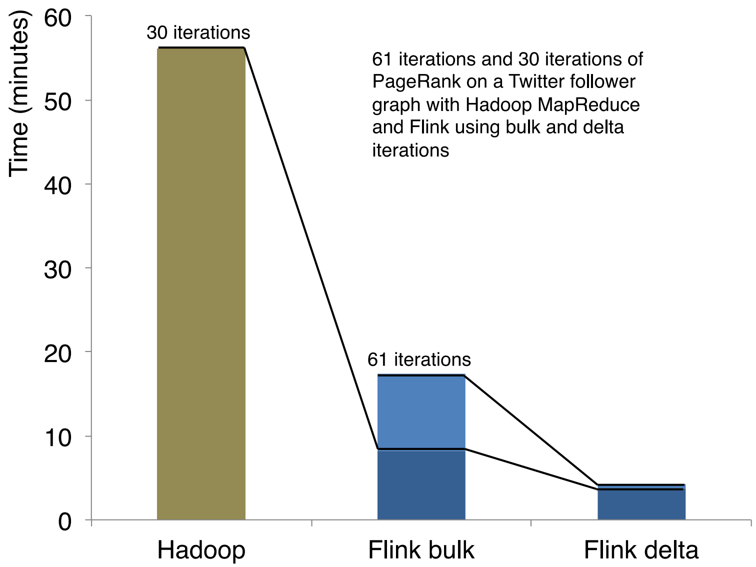
Effect of delta iterations…



… very fast graph analysis

Performance competitive with dedicated graph analysis systems

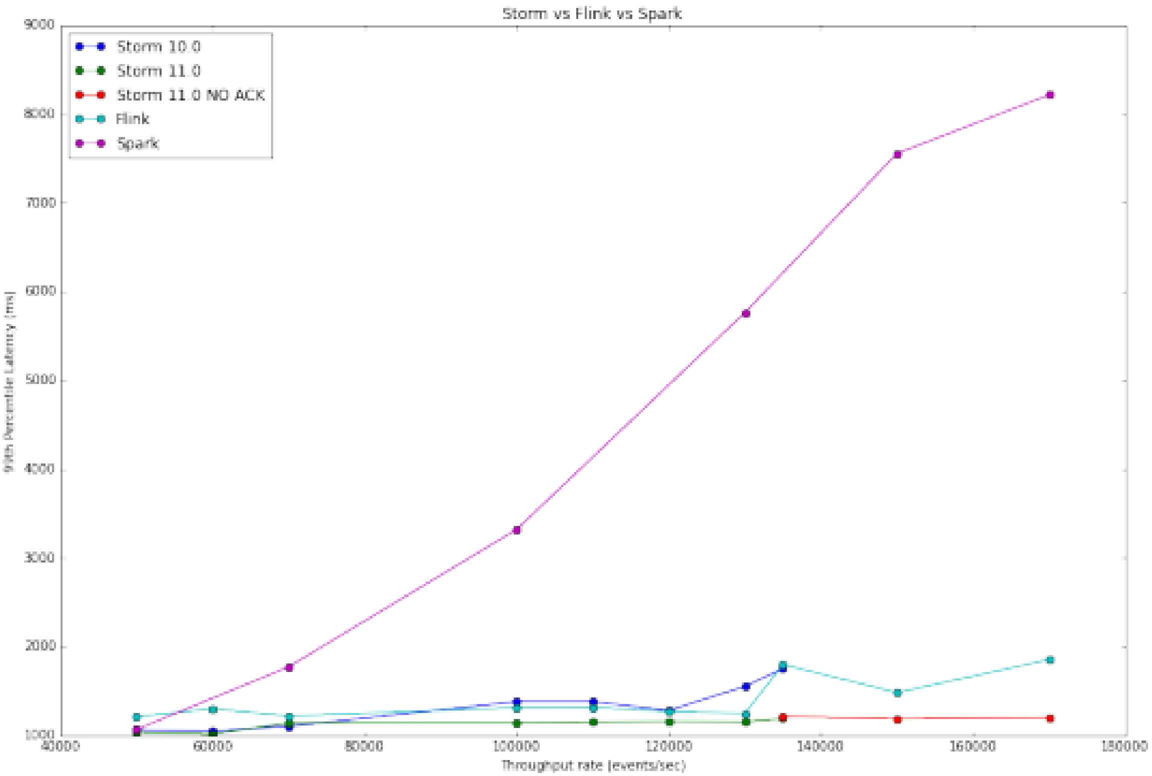




… and mix and match ETL-style and graph analysis in one program

More at: http://data-artisans.com/data-analysis-with-flink.html

Current Benchmark Results

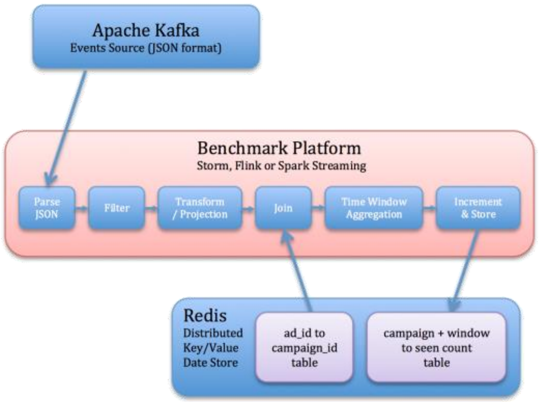


Source: http://yahooeng.tumblr.com/post/135321837876/benchmarking-streaming-computation-engines-at

Performed by Yahoo! Engineering, Dec 16, 2015

[..]Storm 0.10.0, 0.11.0-SNAPSHOT and Flink 0.10.1 show sub- second latencies at relatively high throughputs[..]. Spark streaming 1.5.1 supports high throughputs, but at a relatively higher latency.

Flink achieves highest throughput with competitive low latency!



Timeline

2008
Initial vision for a big data analytics platform

2009
DFG Proposal for Stratosphere I

2010
Grant Award
Start of Stratosphere I

2010
Nephele / PACTs paper published

2011
DFG Proposal for Stratosphere II

2012
Spinning Fast Iterative Dataflows paper published
VLDB

2012
Grant Award
Start of Stratosphere II

2014
Apache Flink Incubator Project

2014
The VLDB Journal
Stratosphere System paper published

2015
Apache Flink Top Level Project

2015
dataArtisans Founded

2015
1st Flink Forward Conference
Flink Forward Berlin 12/13 Oct 2015

2016
Flink Community Groups Across Europe

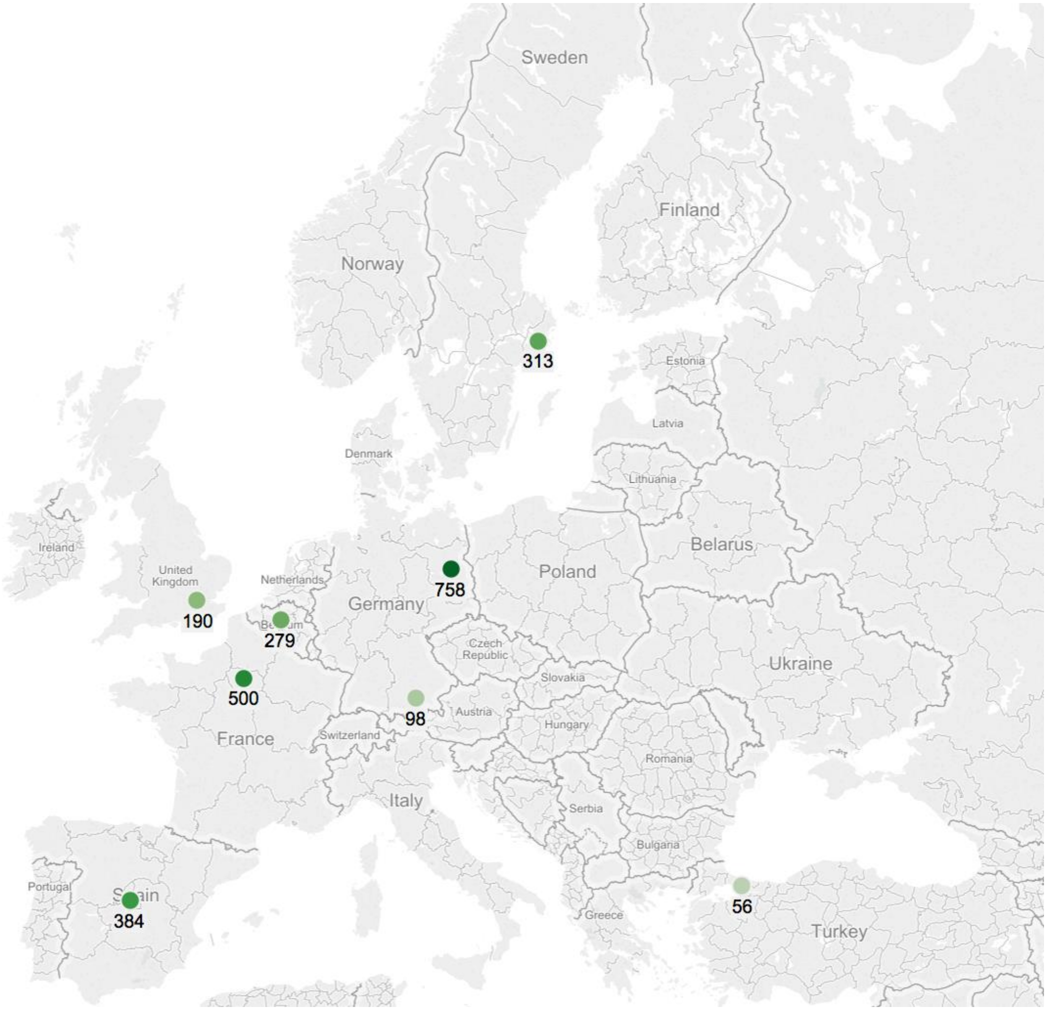
Berlin	758
Paris	500
Madrid	384
Stockholm	313
Brussels	279
London	190

2016
21 Meetups Worldwide

186 Contributors
6326 Members
20 Cities
12 Countries

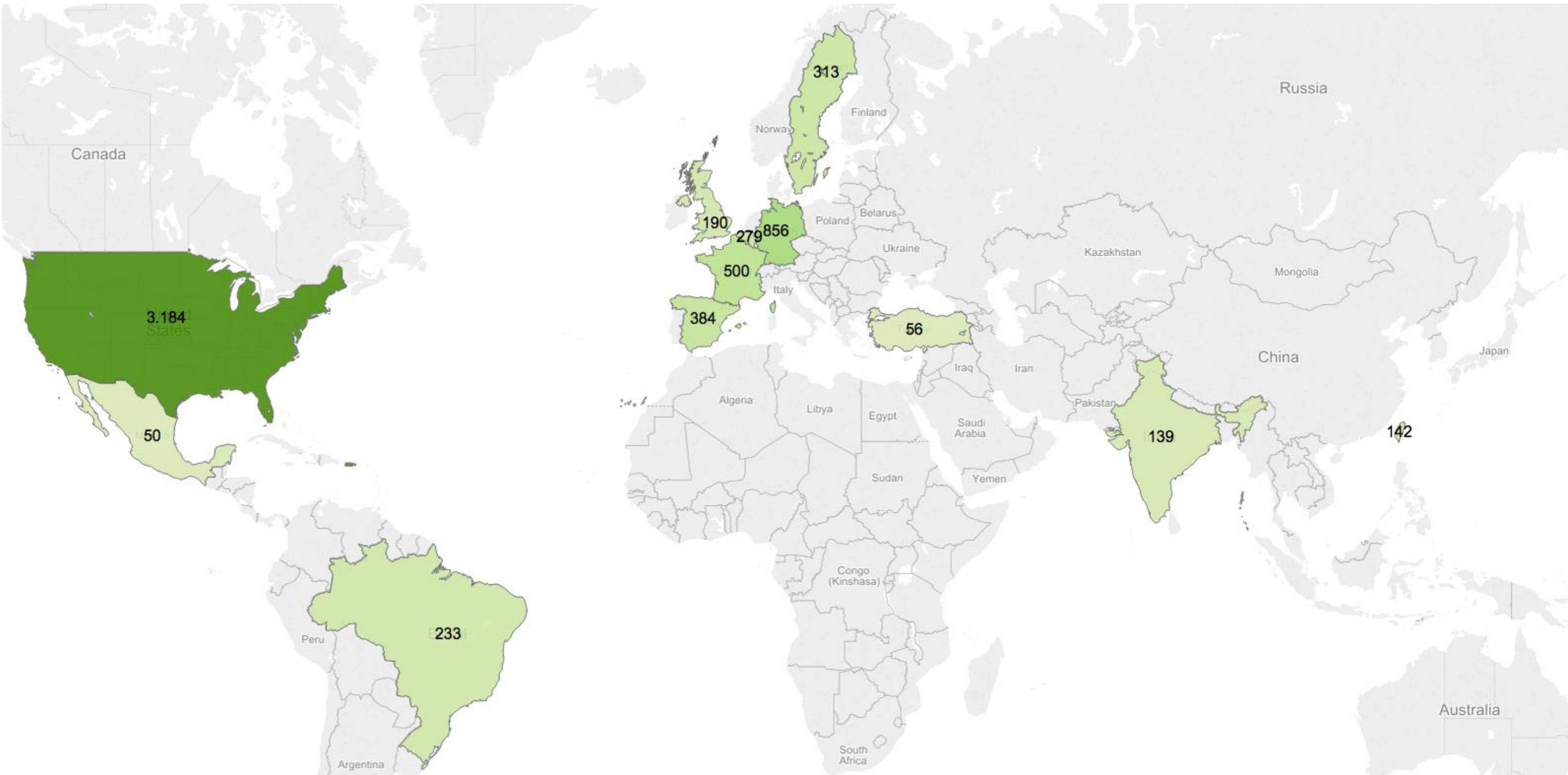
2016
FlinkForward
2nd Flink Forward Conference September 2016
flink-forward.org

(Strictly) Flink European Meetups with Member Totals (as of 30.5.16)



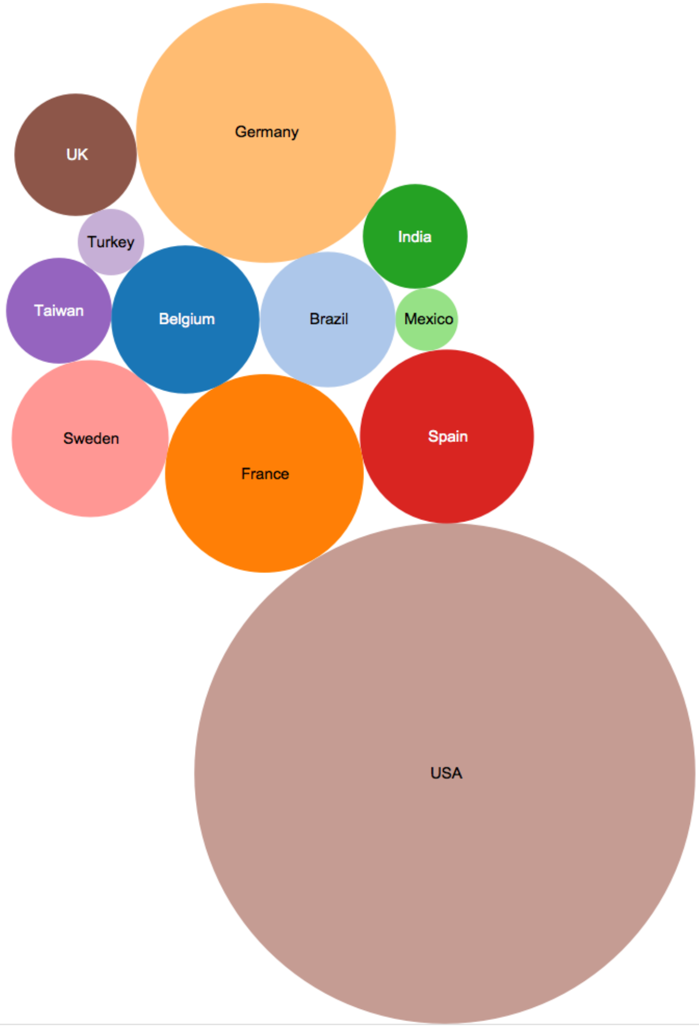
Country	Total Members
Berlin	758
Paris	500
Madrid	384
Stockholm	313
Brussels	279
London	190
Munich	98
Istanbul	56

Meetups By Country Concerning Flink



Apache Flink Meetups Worldwide (Data accurate as of **30.5.16**)
6326 members *strictly focused on* Apache Flink (comprising 57%)
4771 members *broader in scope*, including Flink (comprising 43%)

Distribution of (Strictly) Flink Meetup Group Members by Country (as of 30.5.16)



Country	Total Members
USA	3184
Germany	856
France	500
Spain	384
Sweden	313
Belgium	279
Brazil	233
UK	190
Taiwan	142
India	139
Turkey	56
Mexico	50

> 13 Companies Using Flink

ResearchGate

bouygues TELECOM

OKKAM

Capital One

amadeus

King

treelogic

otto group

zalando

RADICALBIT
Deep, Different, Disruptive.

FIRELAYERS

EURA NOVA

dataArtisans

> 6 Software Projects Using Flink

Google Cloud Platform

CLOUD DATAFLOW

A fully-managed cloud service and programming model for batch and streaming big data processing.

CASCADING

Apache Flink is a replacement for MapReduce to support large-scale batch workloads and streaming data flows. It eliminates the concept of mapping and reducers and leverages in-memory storage, resulting in significant performance gains over MapReduce.

Apache SAMOA

Scalable Advanced Massive Online Analysis

Apache incubator

Apache SAMOA is a distributed streaming machine learning (ML) framework that contains a programing abstraction for distributed streaming ML algorithms.

mahout

The Apache Mahout™ project's goal is to build an environment for quickly creating scalable performant machine learning applications.

Apache MRQL

MRQL is a query processing and optimization system for large-scale, distributed data analysis, built on top of Apache Hadoop, Hama, Spark, and Flink.

beam

Apache Beam is an open source, unified programming model that you can use to create a data processing **pipeline**.

> 10 Research Institutions Using Flink

University of Zagreb

ZIB

BIG DATA EUROPE
Empowering Communities with Data Technologies

HUMBOLDT-UNIVERSITÄT ZU BERLIN

UNIVERSITÄT LEIPZIG

SWEDISH ICT SICS

KTH VETENSKAP OCH KONST

BBDC
BERLIN BIG DATA CENTER

Technische Universität Berlin

DFKI German Research Center for Artificial Intelligence

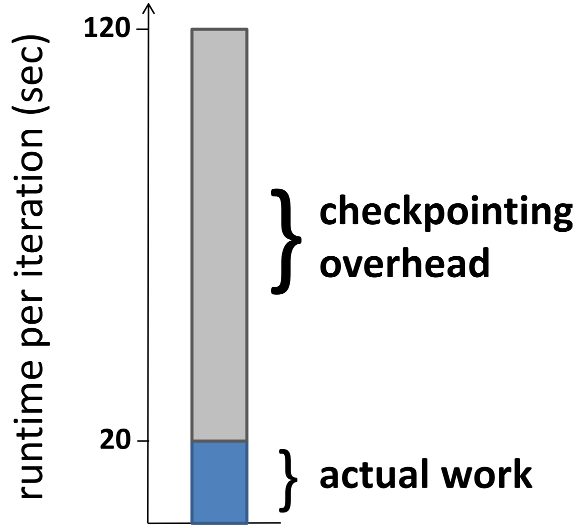
Fault tolerance

Pessimistic Recovery:

- Write intermediate state to stable storage
- Restart from such a checkpoint in case of a failure

Problematic:

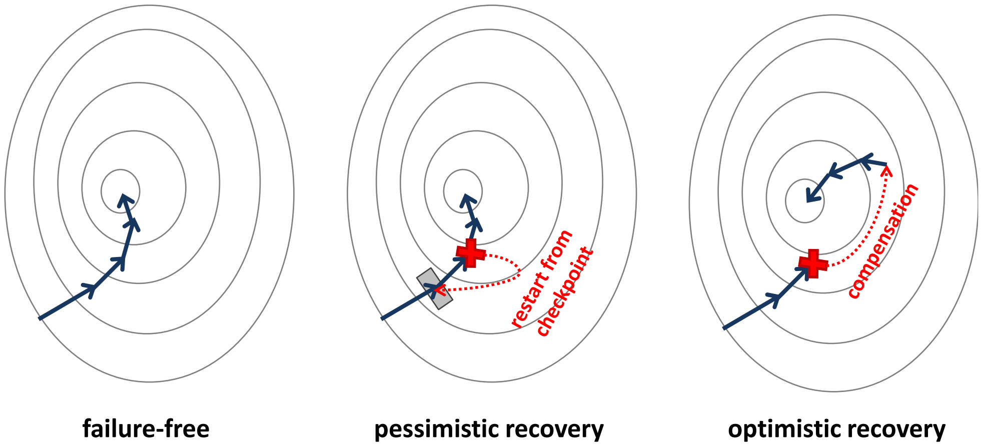
- High overhead, checkpoint must be replicated to other machines
- Overhead always incurred, even if no failures happen!



➢ **How can we avoid this overhead in failure-free cases**

Optimistic recovery

- Many data mining algorithms are **fixpoint algorithms**
- **Optimistic Recovery**: jump to a different state in case of a failure, still converge to solution



- No checkpoints → **No overhead in absense of failures**!
- algorithm-specific **compensation function** must restore state

All Roads lead to Rome

If you are interested more, read our CIKM 2013 paper:

Sebastian Schelter, Stephan Ewen, Kostas Tzoumas, Volker Markl: **"All roads lead to Rome": optimistic recovery for distributed iterative data processing**. *CIKM 2013*: 1919-1928

Sergey Dudoladov, Chen Xu, Sebastian Schelter, Asterios Katsifodimos, Stephan Ewen, Kostas Tzoumas, Volker Markl: **Optimistic Recovery for Iterative Dataflows in Action**. To appear in *SIGMOD 2015*

Declarative Data Processing and Big Data

A Billion $$$ Mantra...

SQL | Relations | RDBMS

Declarative Data Processing

An effective, formal foundation based on relational algebra and calculus (Codd ’71).

A simple, high-level language for querying data (Chamberlin ’74).

An efficient, low-level execution environment tailored towards the data (Selinger ’79).

With 40+ years of success...

SQL

Relations

RDBMS

Declarative Data Processing

Is Being Revised

SQL | Relations | RDBMS

Declarative Data Processing

Second-Order Functions | Distributed Collections | Parallel Dataflow Engines

Overall Vision & Next Steps

- First results
  - Alexander Alexandrov, Andreas Kunft, Asterios Katsifodimos, et al: Implicit Parallelism through Deep Language Embedding. SIGMOD Conference 2015: 47-61
  - Alexander Alexandrov, Andreas Salzmann, Georgi Krastev, Asterios Katsifodimos, Volker Markl: Emma in Action: Declarative Dataflows for Scalable Data Analysis. SIGMOD 2016
  - Alexander Alexandrov, Asterios Katsifodimos, Georgi Krastev, Volker Markl: Implicit Parallelism through Deep Language Embedding. SIGMOD Record 45(1): 51-58 (2016)

- Next Steps (Fall 2016)
  - Open-Source Release

- Vision (Frontend): Multi-model DSL based on type contracts
  - Collection Processing *DataBag[A]*
  - Linear Algebra *Matrix[A], Vector[A]*
  - Stream Processing *Stream[A]*

- Vision (Backend): Target more execution engines
  - Column Stores
  - GPUs

Thanks to my team members and students

- Dr. Stephan Ewen
- Sebastian Schelter
- Dr. Kostas Tzoumas
- Dr. Asterios Katsifodimos
- Fabian Hüske
- Alexander Alexandrov
- Max Heimel

and many more members of the Stratosphere Project, the Berlin Big Data Center, and the Apache Flink community

Evolution of Big Data Platforms