



Distributed Computing and Big Data: Hadoop and MapReduce

Bill Keenan, Director
Terry Heinze, Architect

Thomson Reuters Research & Development



THOMSON REUTERS

Agenda

- R&D Overview
- Hadoop and MapReduce Overview
- Use Case: Clustering Legal Documents

Thomson Reuters

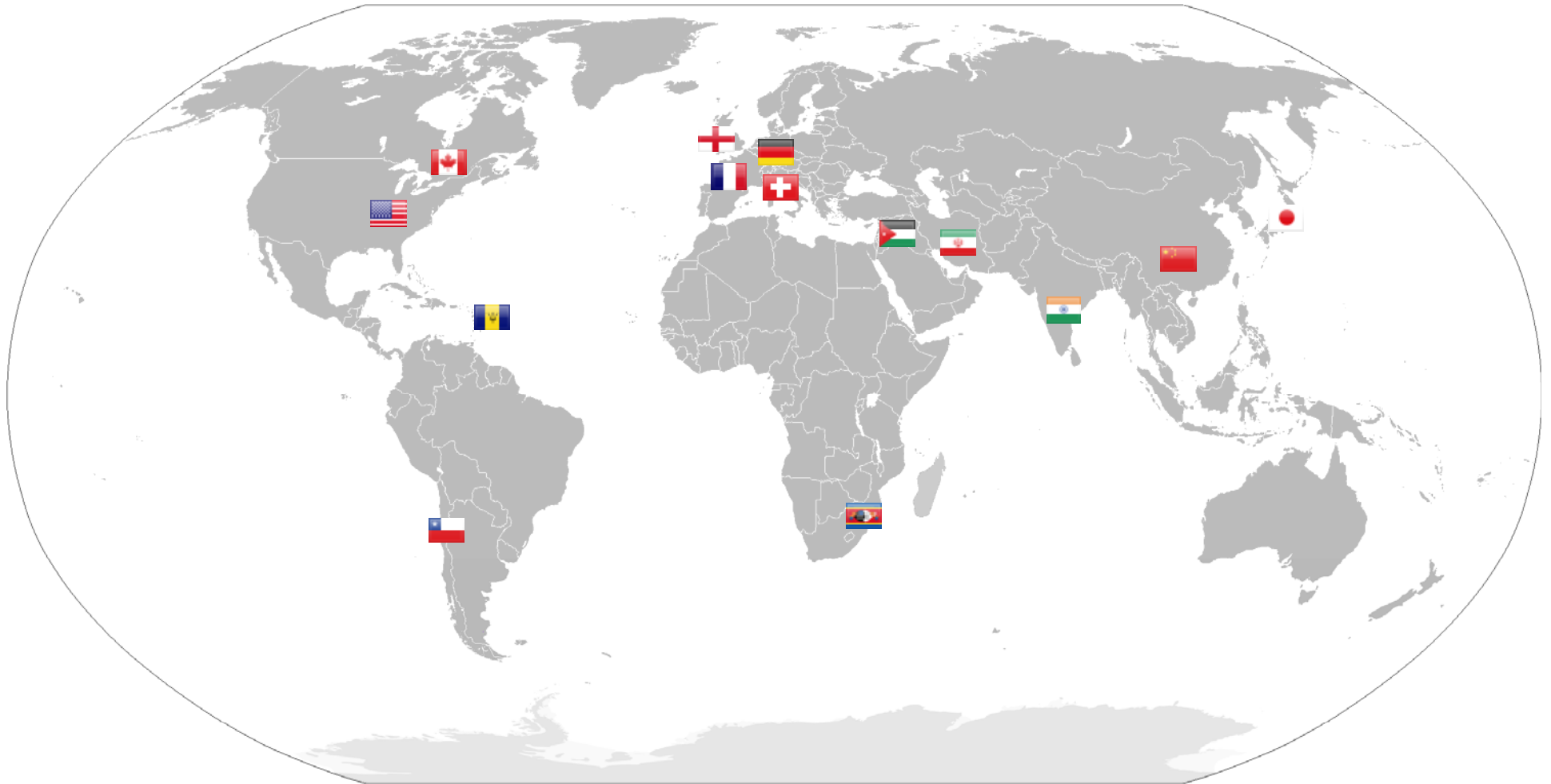
- Leading source of intelligent information for the world's businesses and professionals.
- 55,000+ employees across more than 100 countries
- Financial, Legal, Tax and Accounting, Healthcare, Science and Media markets
- Powered by the world's most trusted news organization (Reuters).

Overview of Corporate R&D

- 40+ computer scientists
 - Research scientists, Ph.D. or equivalent
 - Software engineers, architects, project managers
- Highly focused areas of expertise
 - Information retrieval, text categorization, financial research
 - Financial analysis
 - Text & data mining, machine learning
 - Web service development, Hadoop



Our International Roots



Role Of Corporate R&D



Anticipate

Research



Partner

Deliver





Hadoop and MapReduce



THOMSON REUTERS

Big Data and Distributed Computing

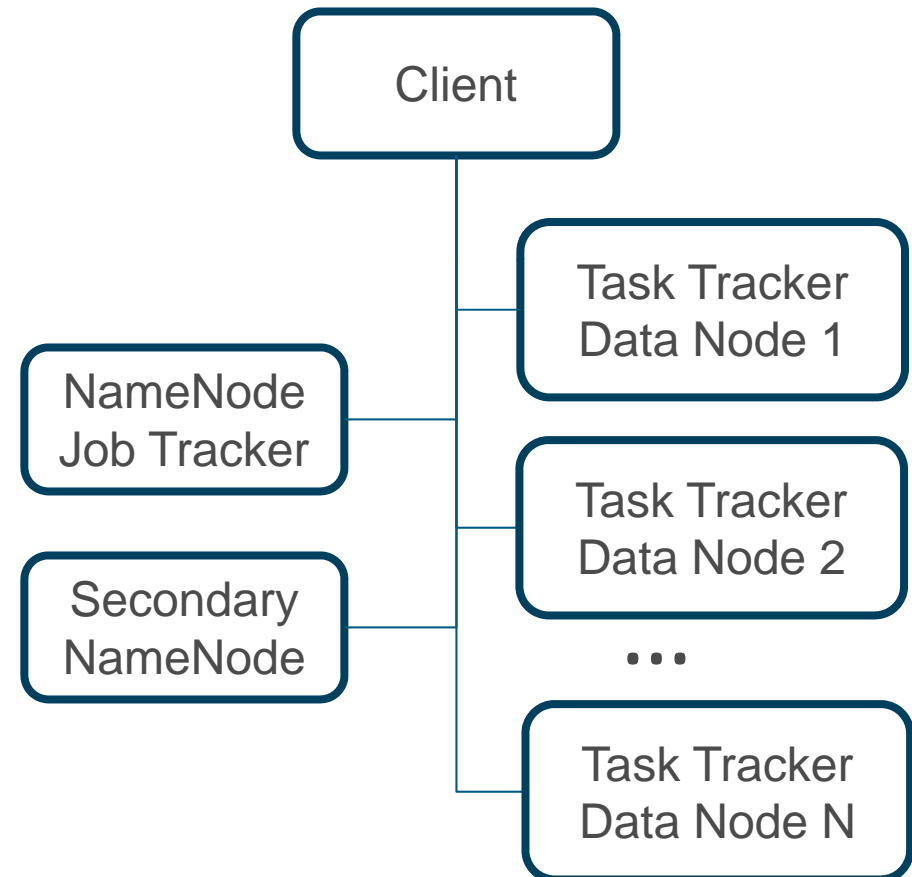
- Big Data at Thomson Reuters
 - More than 10 petabytes in Eagan alone
 - Major data centers around globe: financial markets, tick history, healthcare, public records, legal documents
- Distributed Computing
 - Multiple architectures and use cases
 - Focus today: using multiple servers, each working on part of job, each doing same task
 - Key Challenges:
 - Work distribution and orchestration
 - Error recovery
 - Scalability and management

Hadoop & MapReduce

- **Hadoop:** A software framework that supports distributed computing using MapReduce
 - Distributed, redundant file system (HDFS)
 - Job distribution, balancing, recovery, scheduler, etc.
- **MapReduce:** A programming paradigm that is composed of two functions (~ relations)
 - Map
 - Reduce
 - Both are quite similar to their functional programming cousins
- Many add-ons

Hadoop Clusters

- NameNode: stores location of all data blocks
- Job Tracker: work manager
- Task Tracker: manages tasks on one Data Node
- Client accesses data on HDFS, sends jobs to Job Tracker



HDFS Key Concepts

- Google File System
- Small # of large files
- Streaming batch processes
- Redundant, rack aware
- Failure resistant
- Write-once (usually), read many
- Single point failure
- Incomplete security
- Not only for MapReduce

Map/Reduce Key Concepts

- <key,value>
- Mappers: input -> intermediate kv pairs
- Reducers: intermediate -> output kv pairs
- InputSplits
- Progress reporting
- Shuffling, partitioning
- Scheduling
- Task distribution
- Topology aware
- Distributed cache
- Recovery
- Compression
- Bad Records
- Speculative execution

Use Cases

- Query log processing
- Query mining
- Text Mining
- XML transformations
- Classification
- Document Clustering
- Entity Extraction

Case Study: Large Scale ETL

- Big Data: Public Records
- Warehouse loading long process, expensive infrastructure, complex management
- Combine data from multiple repositories (extract, transform, load)
- Idea:
 - Use Hadoop's natural ETL capabilities
 - Use existing shared infrastructure

Why Hadoop

- Big data – billions of documents
- Needed to process each document, combine information
- Expected multiple passes, multiple types of transformations
- Minimal workflow coding

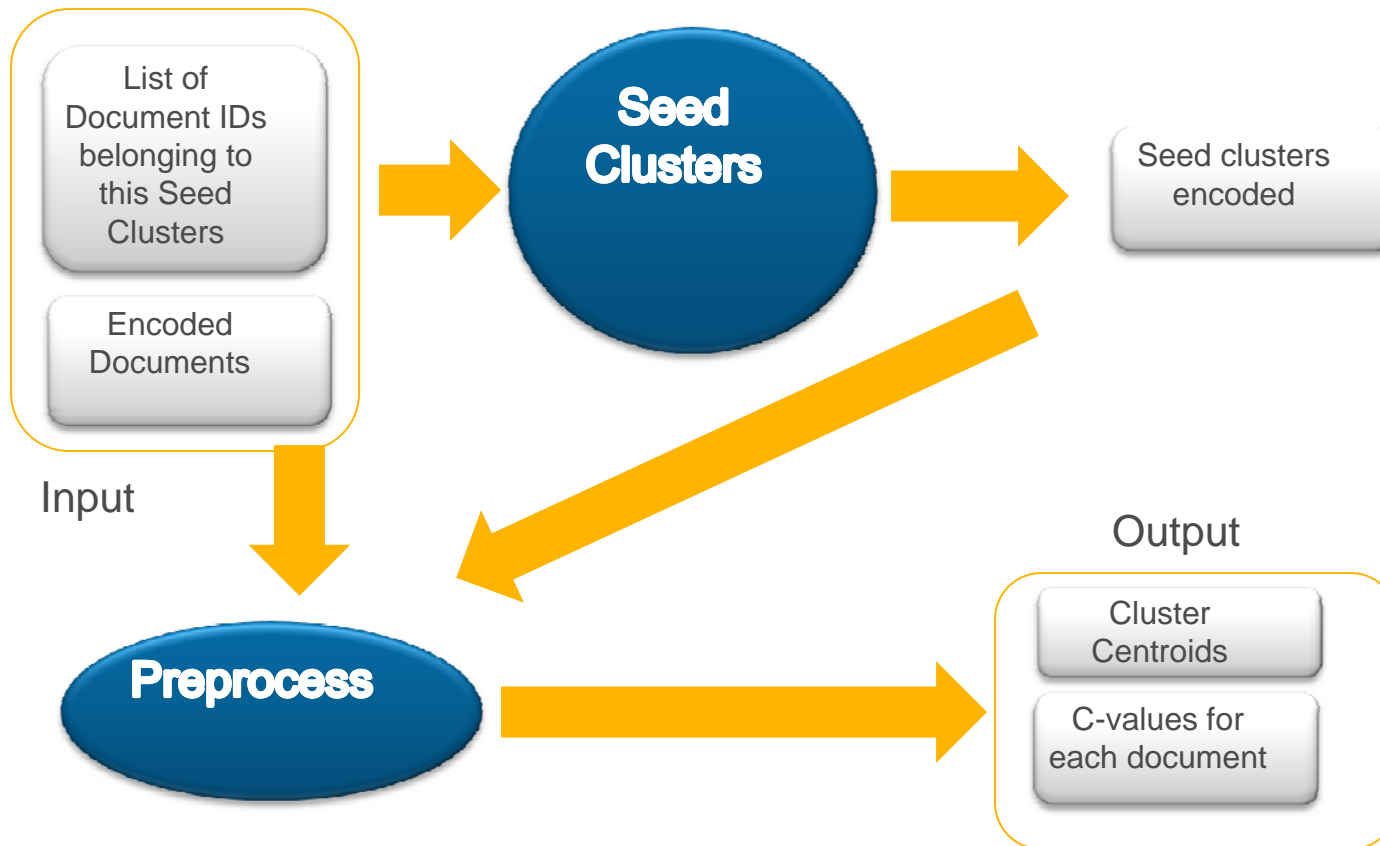
Use Case: Language Modeling

- Build Languages Models from clusters of legal documents
- Large initial corpus: 7,000,000 xml documents
- Corpus grows over time

Process

- Prepare the input
 - Remove duplicates from the corpus
 - Remove stop words (common English, high frequency terms)
 - Stem
 - Convert to binary (sparse TF vector)
 - Create centroids for seed clusters

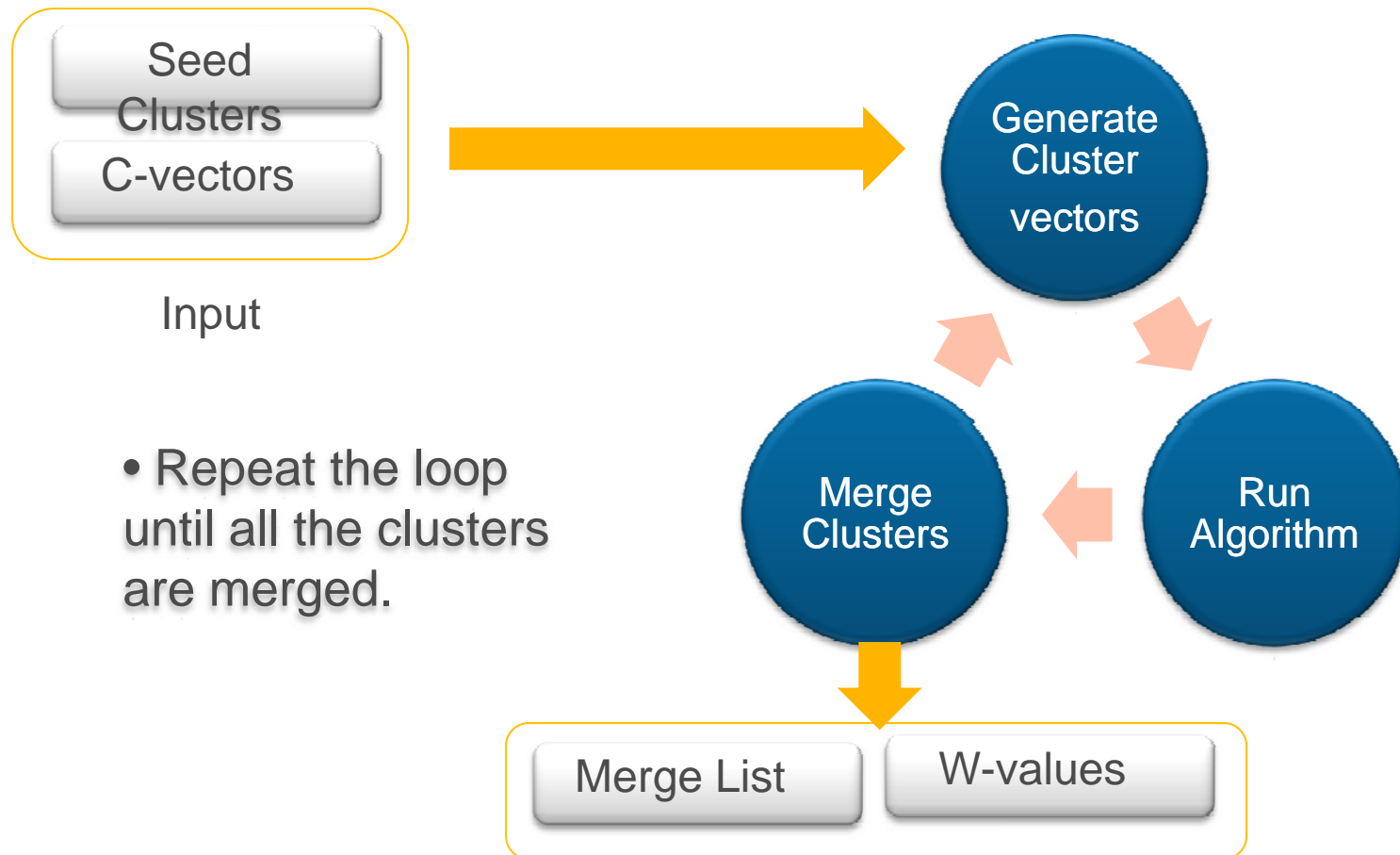
Process



Process

- Clustering
 - Iterate until number of clusters equals goal
 - Multiply matrix of document vectors and matrix of cluster centroids
 - Assign document to best cluster
 - Merge clusters and re-compute centroids

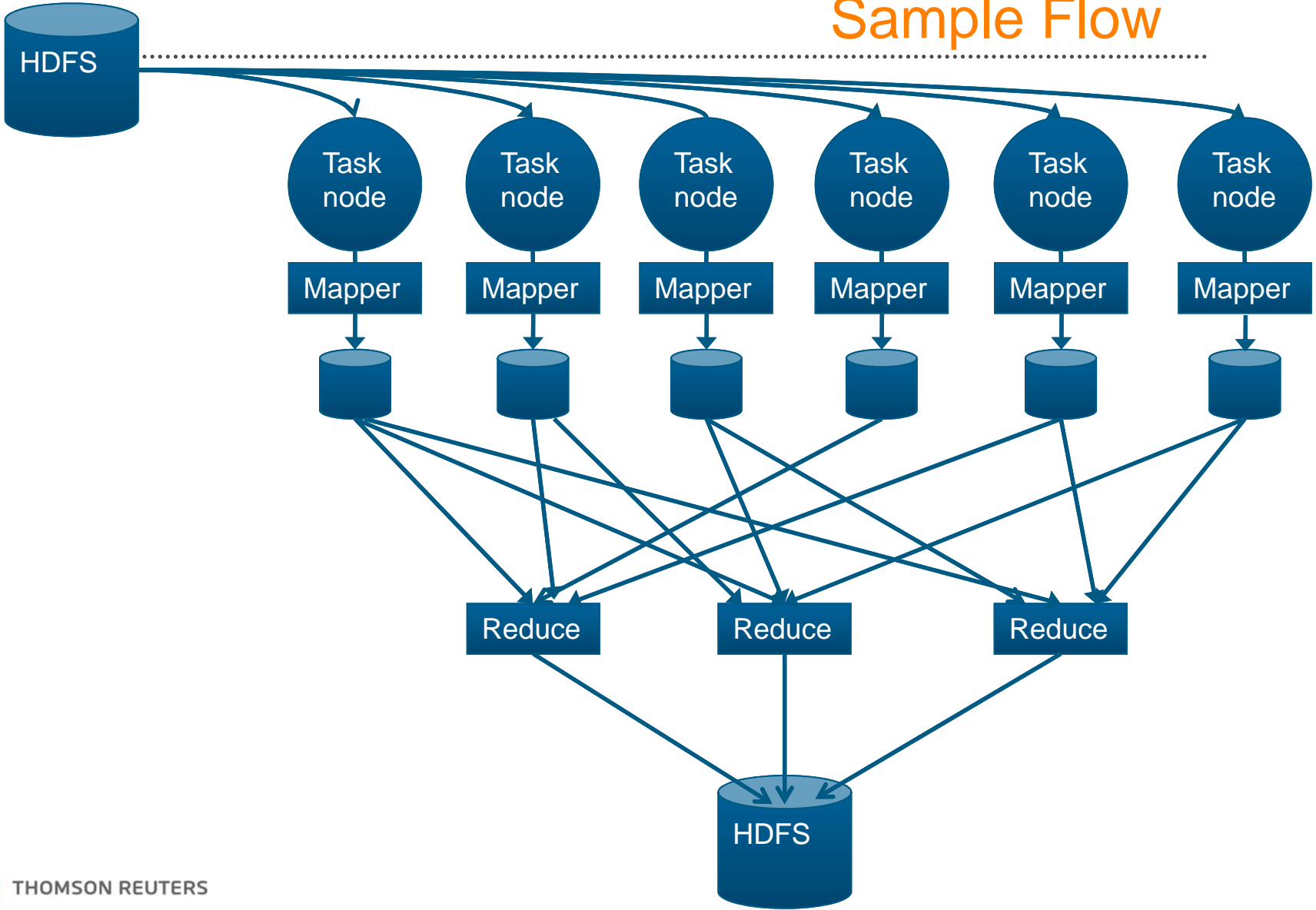
Process



Process

- Validate and Analyze Clusters
 - Create classifier from clusters
 - Assign all non-clustered documents to clusters using the classifier
 - Build Language Model for each cluster

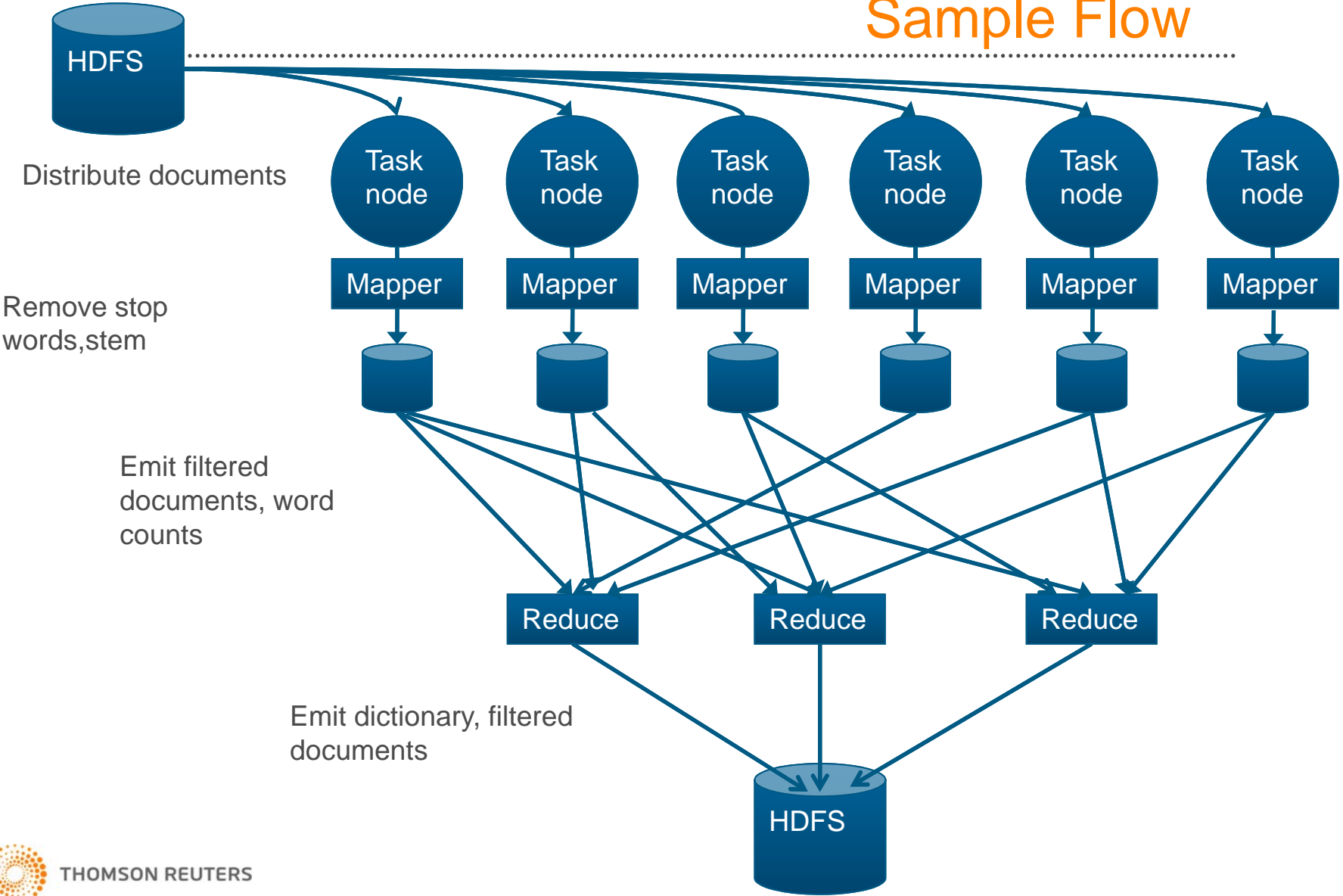
Sample Flow



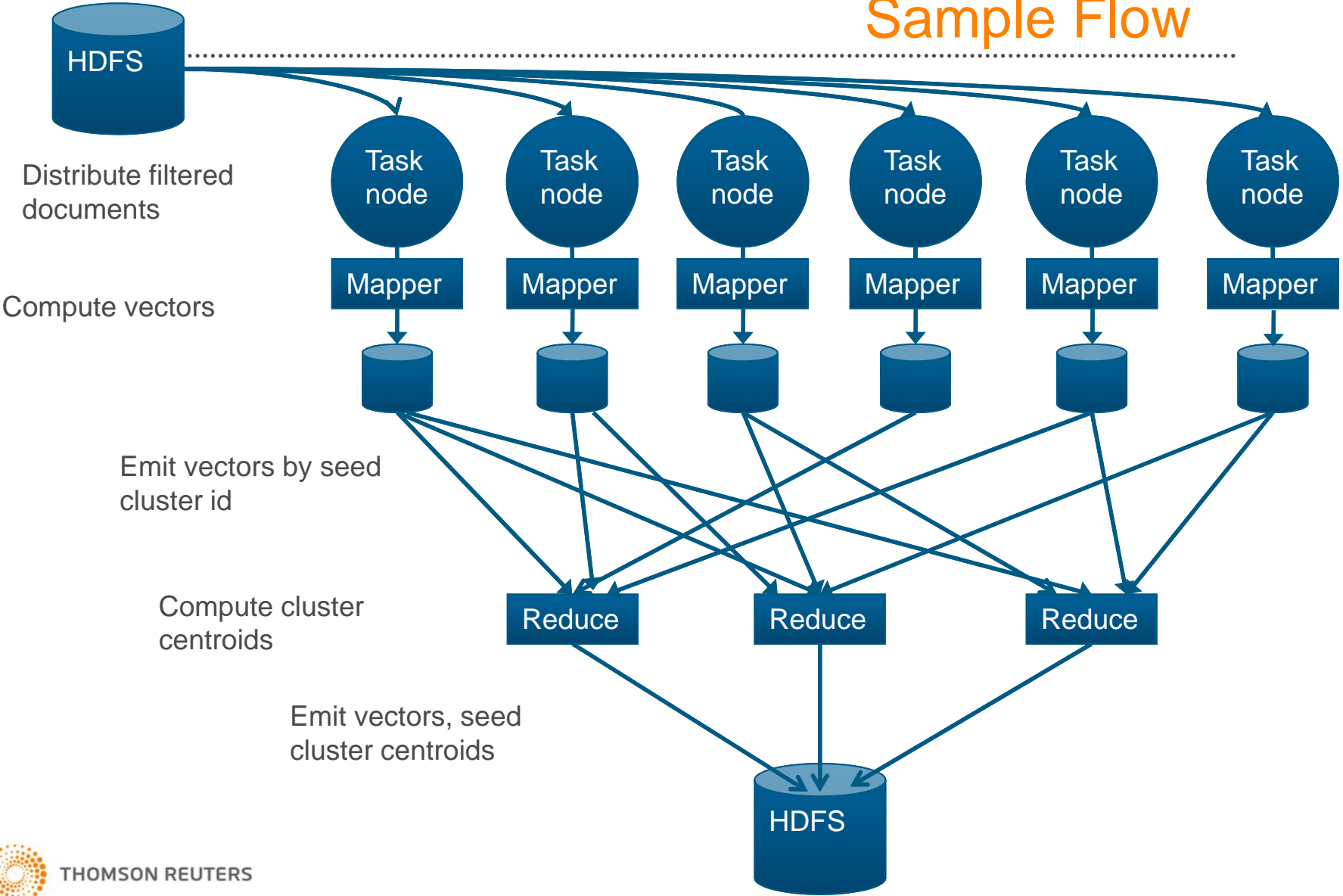
Prepare Input using Hadoop

- Fits the Map/Reduce paradigm
 - Each document is atomic: documents can be equally distributed within the HDFS
 - Each mapper removes stop words, tokenizes, and stems
 - Mappers emit token counts, hashes, and tokenized documents
 - Reducers build Document Frequency dictionary (basically, the “word count” example)
 - Reduces the hashes to a single document (de-duplication)
 - Additional Map/Reduce converts tokenized documents to sparse vectors using the DF dictionary
 - Additional MapReduce maps document vectors and seed cluster ids and reducer generates centroids

Sample Flow



Sample Flow



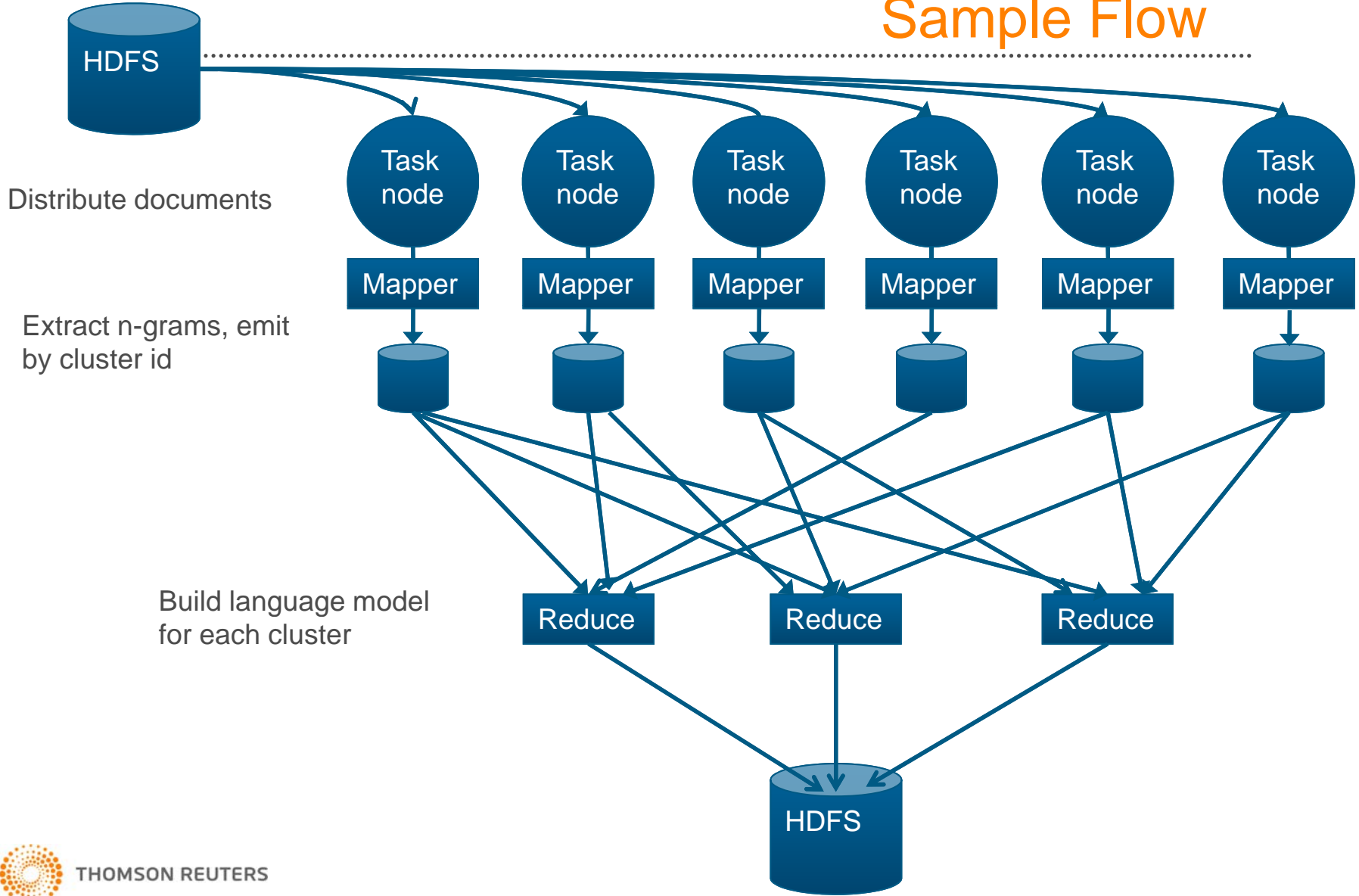
Clustering using Hadoop

- Map/Reduce paradigm
 - Each document vector is atomic: documents can be equally distributed within the HDFS
 - Mapper initialization required loading large matrix of cluster centroids
 - Large memory utilization to hold matrix multiplications
 - Decompose matrices into smaller chunks and run multiple map/reduce steps to obtain final result matrix (new clusters)

Validate and Analyze Clusters using Hadoop

- Map/Reduce paradigm
 - A document classifier based on the documents within the clusters was built
 - n.b. the classifier itself was trained using Hadoop
 - Un-clustered documents (still in the HDFS) are classified in a mapper and assigned a cluster id.
 - A reduction step then takes each set of original documents in a cluster and creates a language model for each cluster

Sample Flow



Using Hadoop

- Other Experiments

- WestlawNext Log Processing

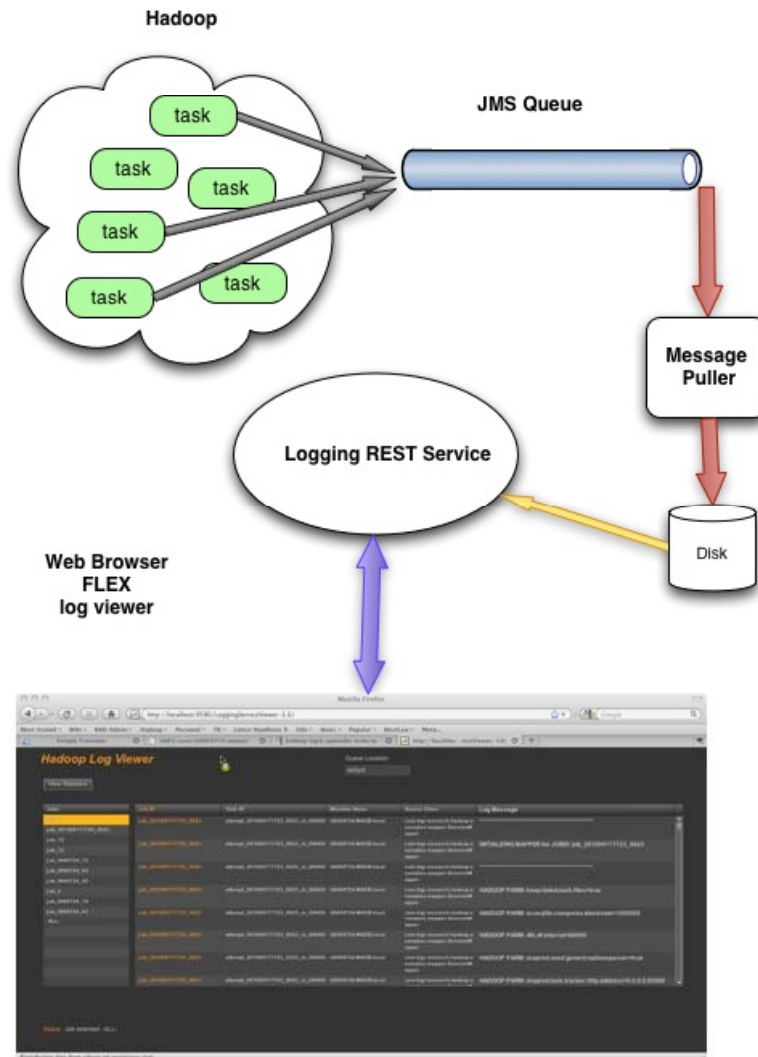
- Billions of raw usage events are generated
 - Used Hadoop to map raw events to a user's individual session
 - Reducers created complex session objects
 - Session objects reducible to xml for xpath queries for mining user behavior

- Remote Logging

- Provide a way to create and search centralized Hadoop job logs, by host, job, and task ids
 - Send the logs to a message queue
 - Browse the queue or...
 - Pull the logs from the queue and retain them in a db



Remote Logging: Browsing Client



Lessons learned

- State of Hadoop
 - Weak security model, changes in works
 - Cluster configuration, management and optimization still sometimes difficult
 - Users can overload a cluster. Need to balance optimization and safety.
- Learning curve moderate
 - Quick to run first naïve MR programs
 - Skill/experience required for advanced or optimized processes

Lessons Learned

- Loading HDFS is time consuming: **Wrote multi-threaded loader to reduce bound IO**
- Multiple step process needed to be re-run using different test corpuses: **Wrote parameterized Perl script to submit jobs to the Hadoop cluster**
- Test Hadoop on a single node cluster first: **Install Hadoop locally**
 - Local mode within Eclipse (Windows, Mac)
 - Pseudo-distributed mode (Mac, Cygwin, VMWare) using Hadoop plugin (Karmasphere)



Lessons Learned

- Tracking intermediate results: Detect bad or inconsistent results after each iteration
 - Record messages to Hadoop node logs
 - Create remote logger (event detector) to broadcast status
- Regression tests: Small, sample corpus run through local Hadoop and distributed Hadoop. Intermediate and final results compared against reference results created by baseline Matlab application



Lessons Learned

- Performance evaluations
 - Detect bad or inconsistent results after each iteration
 - Don't accept long duration tasks as “normal”
 - Regression test on Hadoop took 4 hours while same Matlab test took seconds (because of the need to spread the matrix operations over several map/reduce steps)
 - Re-evaluated core algorithm and found ways to eliminate and compress steps related to cluster merging
 - Direct conversion of mathematics, as developed, to java structures and map/reduce was not efficient
 - New clustering process no longer uses Hadoop: 6 hours on single machine vs. 6 days on 20 node Hadoop cluster
 - As size corpus grows, we will need to migrate new cluster algorithm back to Hadoop



Lessons Learned

- Performance evaluations
 - Leverage combiners and mapper statics
 - Reduce the amount of data during shuffle

Lessons Learned

- Releases are still volatile
 - Core API changed significantly from release .19 to .20
 - Functionality related to distributed cache changed (application files loaded to each node at runtime)
 - Eclipse Hadoop plugins
 - Source code only with release .20 and only works in older Eclipse versions on Windows
 - Karmasphere plugin (Eclipse and NetBeans) more mature but still more of a concept than productive
 - Just use Eclipse for testing Hadoop code in local mode
 - Develop alternatives for handling distributed cache

Reading

Hadoop

The Definitive Guide

Tom White
O'Reilly

Data-Intensive Text Processing with MapReduce

Jimmy Lin and Chris Dyer
University of Maryland
<http://www.umiacs.umd.edu/~jimmylin/book.html>



THOMSON REUTERS

Questions?

Thank you.