Concepts and Models of Parallel and Data-centric Programming

MapReduce – Comparison to Other Approaches

Lecture, Summer 2020

Simon Schwitanski
Dr. Christian Terboven

i12 High Performance Computing
RWTH AACHEN UNIVERSITY

Outline

0. Organization
1. Foundations
2. Shared Memory
3. GPU Programming
4. Bulk-Synchronous Parallelism
5. Message Passing
6. Distributed Shared Memory
7. Parallel Algorithms
8. Parallel I/O
9. **MapReduce**
10. Apache Spark

a. MapReduce Programming Model
b. Parallelizing MapReduce
c. Hadoop Ecosystem
d. Hadoop Distributed File System
e. Yet Another Resource Negotiator
f. Comparison to Other Approaches
g. MapReduce Design Patterns

MapReduce vs. MPI vs. DBMS

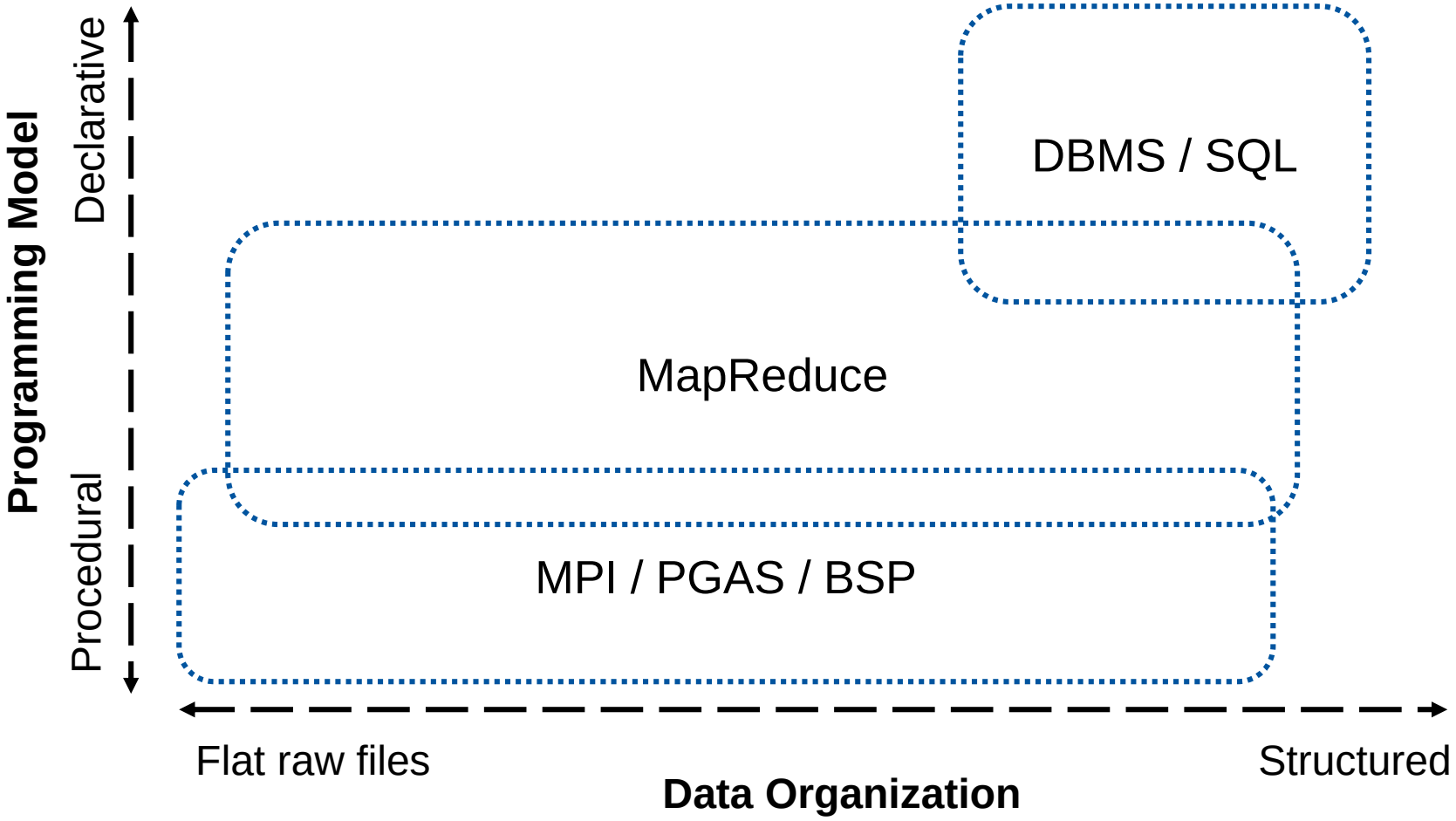


Illustration by Zhao, Jerry and Pjesivac-Grbovic, Jelena. “MapReduce: The Programming Model and Practice”, SIGMETRICS 2009 Tutorials, June 2009, https://static.googleusercontent.com/media/research.google.com/de//pubs/archive/36249.pdf

MapReduce vs. MPI / PGAS / BSP vs. DBMS

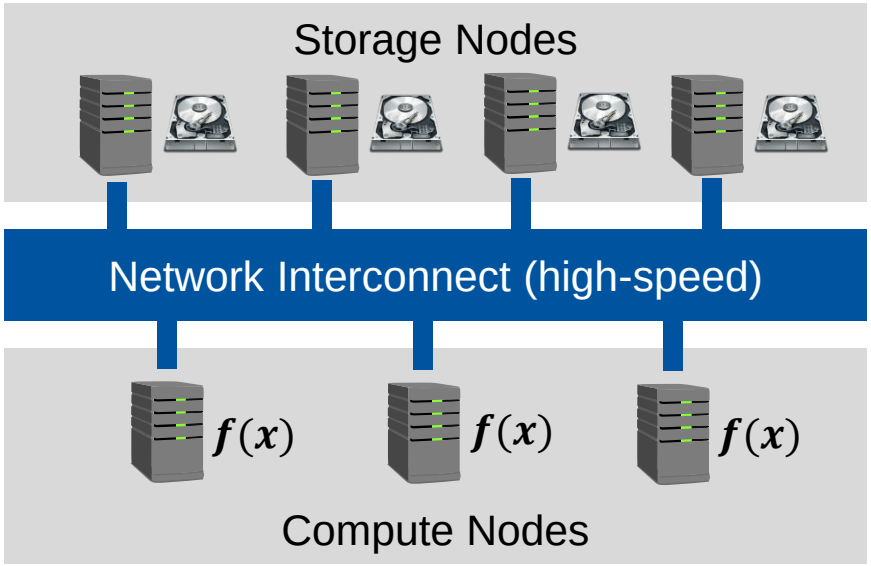
	MPI / PGAS / BSP	MapReduce	DBMS / SQL
General Description	General parallel programming paradigm	Programming paradigm and associated execution system	System to store, manipulate and serve data
Programming Model	Messages passed between nodes	Restricted to Map / Reduce operation	Declarative on data query; stored procedures
Data organization	No assumption	“Files” can be sharded (split in independent pieces)	Organized data structures
Data to be manipulated	Any	KV pairs	Tables with rich types
Execution Model	Nodes are independent (BSP: Supersteps)	Map / Shuffle / Reduce, Data locality	Transaction, Query optimization, Materialized view
Usability	Steep learning curve, difficult to debug	Simple concept, can be hard to optimize	Declarative interface, can be hard to debug in runtime
Key Selling Point	Scalability, adapts to various applications	Process large amount of data with commodity hardware	Interactive querying data, consistent view across clients

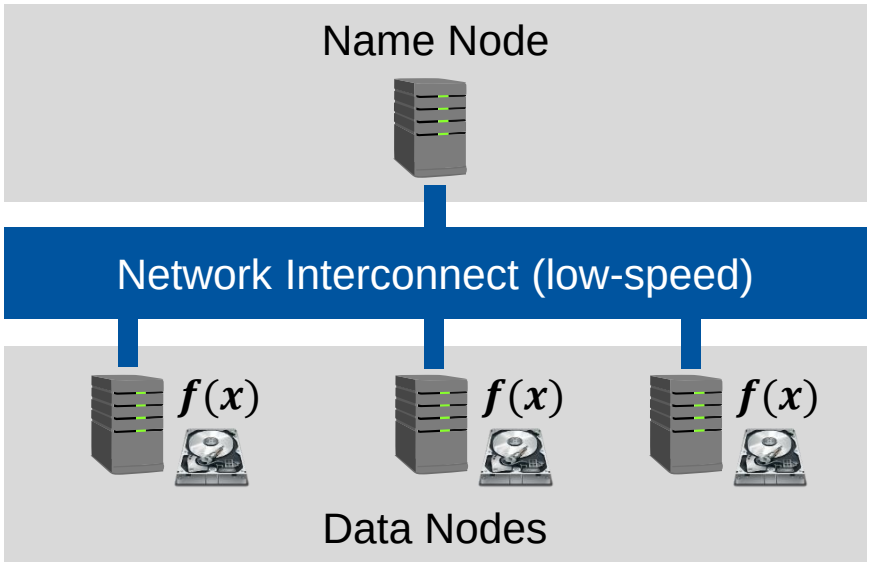
Zhao, Jerry and Pjesivac-Grbovic, Jelena. “MapReduce: The Programming Model and Practice”, SIGMETRICS 2009 Tutorials, June 2009, https://static.googleusercontent.com/media/research.google.com/de//pubs/archive/36249.pdf

Infrastructure: Supercomputer vs. Big Data Cluster

	Traditional Supercomputer	Big Data Cluster
Hardware	Strongly specialized for compute-intensive jobs	Commodity hardware
Interconnect	Low-latency interconnects: InfiniBand, Intel OmniPath	Commodity interconnect: Ethernet (1 / 10 GbE)
File System	Compute and storage nodes physically separated	Locally attached storage at each node
Software Stack	Specialized on scientific computing	Specialized on data science computing
Typical Workload	Computation-intensive “Move data to computation”	Data-intensive “Move computation to data”

File System Comparison





Lustre, GPFS (Supercomputers)

- Separate storage and computation
- Compute nodes retrieve data from storage nodes via interconnect
- High-speed interconnect efficient even for data-intensive jobs

HDFS (Big Data Clusters)

- Focus on data locality
- Each data node: Locally attached storage
- Datasets distributed among nodes
- Job assignments consider data locality

What you have learnt

- MapReduce: Programming model and runtime system
- Well-suited for data-intensive tasks
- Focus on simplicity, scalability and fault tolerance
- Most prominent implementation: Apache Hadoop
  - File system: Hadoop Distributed File System (HDFS)
  - Cluster resource management: Yet Another Resource Negotiator (YARN)
- Comparison with MPI / PGAS / BSP and DBMS / SQL
  - MapReduce lies in between procedural approach of MPI / PGAS / BSP and declarative approach of SQL
- Comparison of traditional supercomputer and big data cluster
  - Highly specialized hardware vs. commodity hardware