

Scalable Scientific Data Processing

B. Gufler J. Müller T. Scholl A. Reiser A. Kemper

Technische Universität München
Department of Computer Science
Garching, Germany

Scalable Scientific Data Processing

Data Analysis

- ▶ exploits massive parallelism
- ▶ supports handling of structured data
- ▶ based on Cloud technology

Scalable Scientific Data Processing

Data Analysis

- ▶ exploits massive parallelism
- ▶ supports handling of structured data
- ▶ based on Cloud technology

Data Management

- ▶ scalable, decentralised approach
- ▶ supports community-specific access patterns
- ▶ based on DHT systems

Data Management: HiSbase

Goal: Scalable storage and retrieval for scientific data sets

- ▶ combines globally distributed data sources
- ▶ data load balancing by partitioning
- ▶ support for community specific data access patterns
 - ▶ e.g., preserve spacial locality for observational catalogues
- ▶ query load balancing through replication
- ▶ uses P2P techniques (DHT, key-based routing)

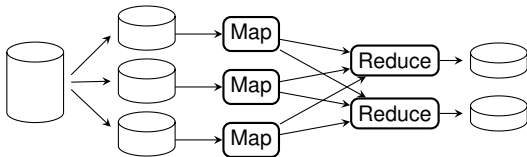
Data Analysis

Goal: Fast and easy analysis of scientific data sets

- ▶ scientific data sets
 - ▶ large scale (TBs per catalogue/simulation)
 - ▶ possibly structured data (e.g. Millennium merger trees)
- ▶ fast analysis
 - ▶ massive parallelism
 - ▶ scalable infrastructure
- ▶ easy analysis
 - ▶ high level scripting language
 - ▶ hide technical details

Data Analysis: Based on MapReduce

- ▶ proposed by Google for processing very large data sets
- ▶ one data set
- ▶ two processing steps
- ▶ file based data exchange via full mesh
- ▶ transparent handling of infrastructure failures



Data Analysis: Pipelined MapReduce

- ▶ permits more than two processing steps
 - ▶ pipelined data exchange via full mesh, broadcast, or 1:1
 - ▶ allows to process multiple data sets
- ⇒ more flexibility for sophisticated workflows

Data Analysis: Based on PigLatin

- ▶ scripting language for MapReduce jobs
- ▶ offers data processing primitives
 - ▶ projection
 - ▶ selection
 - ▶ grouping
 - ▶ aggregation
 - ▶ (equi)join
- ▶ automatically compiled to a sequence of MapReduce tasks

Data Analysis: TreeLatin

- ▶ extends PigLatin to handle hierarchical data sets
- ▶ translates to workflows for Pipelined MapReduce
- ▶ uses database optimisation techniques
 - ▶ pushing selections and projections
 - ▶ early aggregation
 - ▶ different join algorithms

Get In Touch

- ▶ Database systems group, TU München
 - ▶ web: <http://www-db.in.tum.de/>
 - ▶ eMail: gufler@in.tum.de
- ▶ Cloud data analysis project
 - ▶ <http://www-db.in.tum.de/research/projects/cdm>
- ▶ HiSbase project
 - ▶ <http://www-db.in.tum.de/research/projects/hisbase>
- ▶ GAVO
 - ▶ <http://www.g-vo.org>

Thank You for Your Attention