Big Data Ogres and their Facets
Geoffrey Fox, Judy Qiu, Shantenu Jha, Saliya Ekanayake

- **Big Data Ogres** are an attempt to characterize applications and algorithms with a set of general common features that are called **Facets**
- Originally derived from NIST collection of 51 use cases but refined with experience
- The 50 facets capture common characteristics (shared by several problems) which are inevitably multi-dimensional and often overlapping. **Divided into 4 views**
  - One view of an Ogre is the overall **problem architecture** which is naturally related to the machine architecture needed to support data intensive application.
  - The **execution (computational) features** view, describes issues such as I/O versus compute rates, iterative nature and regularity of computation and the classic V’s of Big Data: defining problem size, rate of change, etc.
  - The **data source & style** view includes facets specifying how the data is collected, stored and accessed. Has classic database characteristics
  - **Processing** view has facets which describe types of processing steps including nature of algorithms and kernels e.g. Linear Programming, Learning, Maximum Likelihood
  - **Instances of Ogres** are particular big data problems and a set of Ogre instances that cover enough of the facets could form a comprehensive **benchmark/mini-app** set
  - Ogres and their instances can be **atomic** or **composite**
Benchmarks/Mini-apps spanning Facets

• Look at NSF Dibbs Project, NIST 51 use cases, Baru-Rabl review

• Catalog facets of benchmarks and choose entries to cover “all facets”

• Micro Benchmarks: SPEC, EnhancedDFSIO (HDFS), Terasort, Wordcount, Grep, MPI, Basic Pub-Sub ....

• SQL and NoSQL Data systems, Search, Recommenders: TPC (-C to x–HS for Hadoop), BigBench, Yahoo Cloud Serving, Berkeley Big Data, HiBench, BigDataBench, Cloudsuite, Linkbench
  • includes MapReduce cases Search, Bayes, Random Forests, Collaborative Filtering

• Spatial Query: select from image or earth data

• Alignment: Biology as in BLAST

• Streaming: Online classifiers, Cluster tweets, Robotics, Industrial Internet of Things, Astronomy; BGBenchmark; choose to cover all 5 subclasses

• Pleasingly parallel (Local Analytics): as in initial steps of LHC, Pathology, Bioimaging (differ in type of data analysis)

• Global Analytics: Outlier, Clustering, LDA, SVM, Deep Learning, MDS, PageRank, Levenberg-Marquardt, Graph 500 entries

• Workflow and Composite (analytics on xSQL) linking above
# 6 Data Analysis Architectures

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Example Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) Map Only</td>
<td>Classic Hadoop in classes 1) 2) but not clearly best in class 1)</td>
</tr>
<tr>
<td>(2) Classic MapReduce</td>
<td>BLAST Analysis, Learning, Local Machine Learning, Pleasingly Parallel</td>
</tr>
<tr>
<td>(3) Iterative Map Reduce or Map-Collective</td>
<td>High Energy Physics (HEP), Histograms, Web search, Recommender Engines</td>
</tr>
<tr>
<td>(4) Point to Point or Map-Communication</td>
<td>Expectation maximization, Clustering Linear Algebra, PageRank</td>
</tr>
<tr>
<td>(5) Map Streaming</td>
<td>MapReduce and Iterative Extensions (Spark, Twister)</td>
</tr>
<tr>
<td>(6) Shared memory Map Communicates</td>
<td>Classic MPI, PDE Solvers and Particle Dynamics, Graph Streaming images from Synchrotron sources, Telescopes, IoT</td>
</tr>
</tbody>
</table>

- **Local Analytics**
  - BLAST Analysis
  - Local Machine Learning
  - Pleasingly Parallel

- **MR Basic Statistics**
  - MapReduce
  - High Energy Physics (HEP)
  - Histograms
  - Web search
  - Recommender Engines

- **Iterative**
  - Expectation maximization
  - Clustering Linear Algebra, PageRank

- **Graph**
  - Classic MPI
  - PDE Solvers and Particle Dynamics
  - Graph Streaming images from Synchrotron sources, Telescopes, IoT

- **Streaming**
  - MapReduce and Iterative Extensions (Spark, Twister)
  - MPI, Giraph
  - Apache Storm

- **Shared Memory**
  - Map Communicates
  - Maps are Bolts

**Diagram**: The diagram illustrates the flow of data processing across different architectures, highlighting the relationship between input, output, and various components like maps, reduce, and iterative processes.