Implications of NIST Big Data Application Classification for Benchmarking

IEEE SPEC Research Group on Big Data online meetings
See: http://hpc-abds.org/kaleidoscope/ for details

August 5 2015
Geoffrey Fox
gcf@indiana.edu
School of Informatics and Computing
Digital Science Center
Indiana University Bloomington
NIST Big Data Initiative

Led by Chaitin Baru, Bob Marcus, Wo Chang

And

Big Data Application Analysis
NBD-PWG (NIST Big Data Public Working Group)
Subgroups & Co-Chairs

• There were 5 Subgroups
  – Note mainly industry

• Requirements and Use Cases Sub Group
  – Geoffrey Fox, Indiana U.; Joe Paiva, VA; Tsegereda Beyene, Cisco

• Definitions and Taxonomies SG
  – Nancy Grady, SAIC; Natasha Balac, SDSC; Eugene Luster, R2AD

• Reference Architecture Sub Group
  – Orit Levin, Microsoft; James Ketner, AT&T; Don Krapohl, Augmented Intelligence

• Security and Privacy Sub Group
  – Arnab Roy, CSA/Fujitsu Nancy Landreville, U. MD Akhil Manchanda, GE

• Technology Roadmap Sub Group
  – Carl Buffington, Vistronix; Dan McClary, Oracle; David Boyd, Data Tactics

• See http://bigdatawg.nist.gov/usecases.php
• and http://bigdatawg.nist.gov/V1_output_docs.php
<table>
<thead>
<tr>
<th>Use Case Title</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertical (area)</td>
<td></td>
</tr>
<tr>
<td>Author/Company/Email</td>
<td></td>
</tr>
<tr>
<td>Actors/Stakeholders and their roles and responsibilities</td>
<td></td>
</tr>
<tr>
<td>Goals</td>
<td></td>
</tr>
</tbody>
</table>

**Use Case Description**

<table>
<thead>
<tr>
<th>Current Solutions</th>
<th>Compute/ (System)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Storage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Networking</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Software</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Big Data Characteristics</th>
<th>Data Source (distributed/centralized)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Volume (size)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Velocity (e.g., real time)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variety (multiple datasets, mashup)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variability (rate of change)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Big Data Science (collection, curation, analysis, action)</th>
<th>Veracity (Robustness Issues, semantics)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Visualization</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data Quality (syntax)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data Types</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data Analytics</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Big Data Specific Challenges (Gaps)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Big Data Specific Challenges in Mobility</td>
<td></td>
</tr>
<tr>
<td>Security &amp; Privacy Requirements</td>
<td></td>
</tr>
<tr>
<td>Highlight issues for generalizing this use case (e.g., for ref. architecture)</td>
<td></td>
</tr>
<tr>
<td>More Information (URLs)</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** No proprietary or confidential information should be included

ADD picture of operation or data architecture of application below table

**Use Case Template**

- 26 fields completed for 51 apps
- Government Operation: 4
- Commercial: 8
- Defense: 3
- Healthcare and Life Sciences: 10
- Deep Learning and Social Media: 6
- The Ecosystem for Research: 4
- Astronomy and Physics: 5
- Earth, Environmental and Polar Science: 10
- Energy: 1
- Now an online form 8/5/2015
51 Detailed Use Cases: Contributed July-September 2013
Covers goals, data features such as 3 V’s, software, hardware

- https://bigdatacoursespring2014.appspot.com/course (Section 5)
- Government Operation(4): National Archives and Records Administration, Census Bureau
- Commercial(8): Finance in Cloud, Cloud Backup, Mendeley (Citations), Netflix, Web Search, Digital Materials, Cargo shipping (as in UPS)
- Defense(3): Sensors, Image surveillance, Situation Assessment
- Healthcare and Life Sciences(10): Medical records, Graph and Probabilistic analysis, Pathology, Bioimaging, Genomics, Epidemiology, People Activity models, Biodiversity
- Deep Learning and Social Media(6): Driving Car, Geolocate images/cameras, Twitter, Crowd Sourcing, Network Science, NIST benchmark datasets
- The Ecosystem for Research(4): Metadata, Collaboration, Language Translation, Light source experiments
- Astronomy and Physics(5): Sky Surveys including comparison to simulation, Large Hadron Collider at CERN, Belle Accelerator II in Japan
- Energy(1): Smart grid

26 Features for each use case
Biased to science
<table>
<thead>
<tr>
<th>No.</th>
<th>Use Case</th>
<th>Volume</th>
<th>Velocity</th>
<th>Variety</th>
<th>Software</th>
<th>Analytics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M0172 World Population Scale Epidemiological Study</td>
<td>100TB</td>
<td>Data feeding into the simulation is small but real time data generated by simulation is massive.</td>
<td>Can be rich with various population activities, geographical, socio-economic, cultural variations</td>
<td>Charm++, MPI</td>
<td>Simulations on a Synthetic population</td>
</tr>
<tr>
<td>2</td>
<td>M0173 Social Contagion Modeling for Planning</td>
<td>10s of TB per year</td>
<td>During social unrest events, human interactions and mobility leads to rapid changes in data; e.g., who follows whom in Twitter.</td>
<td>Data fusion a big issue. How to combine data from different sources and how to deal with missing or incomplete data?</td>
<td>Specialized simulators, open source software, and proprietary modeling environments. Databases.</td>
<td>Models of behavior of humans and hard infrastructures, and their interactions. Visualization of results</td>
</tr>
<tr>
<td>3</td>
<td>M0141 Biodiversity and LifeWatch</td>
<td>N/A</td>
<td>Real time processing and analysis in case of the natural or industrial disaster</td>
<td>Rich variety and number of involved databases and observation data</td>
<td>RDMS</td>
<td>Requires advanced and rich visualization</td>
</tr>
<tr>
<td>4</td>
<td>M0136 Large-scale Deep Learning</td>
<td>Current datasets typically 1 to 10 TB. Training a self-driving car could take 100 million images.</td>
<td>Much faster than real-time processing is required. For autonomous driving need to process 1000's high-resolution (6 megapixels or more) images per second.</td>
<td>Neural Net very heterogeneous as it learns many different features</td>
<td>In-house GPU kernels and MPI-based communication developed by Stanford. C++/Python source.</td>
<td>Small degree of batch statistical pre-processing; all other data analysis is performed by the learning algorithm itself</td>
</tr>
<tr>
<td>5</td>
<td>M0171 Organizing large-scale image collections</td>
<td>500+ billion photos on Facebook, 5+ billion photos on Flickr.</td>
<td>over 500M images uploaded to Facebook each day</td>
<td>Images and metadata including EXIF tags (focal distance, camera type, etc), Hadoop Map-reduce, simple hand-written multithreaded tools (ssh and sockets for communication)</td>
<td>Hadoop IndexedHBase &amp; HDFS. Hadoop, Hive, Redis for data management. Python:</td>
<td>Robust non-linear least squares optimization problem. Support Vector Machine</td>
</tr>
<tr>
<td>6</td>
<td>M0160 Truthy</td>
<td>30TB/year compressed data</td>
<td>Near real-time data storage, querying &amp; analysis</td>
<td>Schema provided by social media data source. Currently using Twitter only. We plan to expand</td>
<td>Hadoop IndexedHBase &amp; HDFS. Hadoop, Hive, Redis for data management. Python:</td>
<td>Anomaly detection, stream clustering, signal classification and online-learning; Information diffusion</td>
</tr>
</tbody>
</table>
Online Use Case Form

Are there other data acquisition/access/sharing/management/storage issues?
Specify in text box below:

Analytics Tags - Data Format and Nature of Algorithm used in Analytics
Principal Big Data Use Case Details

Overall questions about the use case covering all facets except security and privacy

Overall project description

http://hpc-abds.org/kaleidoscope/survey/

Use Case Title *
Your choice. Best if at most one line

Domain ("Vertical") *
What application area applies? There is no fixed ontology. See examples of existing use cases. "Health Care" "Social Networking" "Financial" "Energy" are examples

Actors / Stakeholders
Identify relevant stakeholder roles and responsibilities. (Note: Security and privacy roles are survey separate part of this survey.)

Project Goals or Objectives

Security and Privacy

Note that there are aspects of curation, provenance and governance that are not strictly speaking only security and privacy considerations. Refer to other sections of this survey for those aspects.

The S&P questions are grouped as follows:
» Roles
» Personally Identifiable Information
» Covenants and Liability
» Ownership, Distribution, Publication
» Risk Mitigation
» Audit and Traceability
» Data Life Cycle
» Dependencies
» Framework provider S&P
» Application Provider S&P
» Information Assurance / System Health
» Permitted Use Cases

Security, privacy, provenance, governance, curation, and system health.

Roles

Roles may be associated with multiple functions within a big data ecosystem.

Investigator, Lead Analyst, Lead Scientists, Project Leader, Mgr Product Dev, VP Engineering

Identify the role associated with identifying the use case need, requirements and deployment

Investigator Affiliations
Features and Examples
51 Use Cases: What is Parallelism Over?

- **People**: either the users (but see below) or subjects of application and often both
- **Decision makers** like researchers or doctors (users of application)
- **Items** such as Images, EMR, Sequences below; observations or contents of online store
  - **Images** or “Electronic Information nuggets”
  - **EMR**: Electronic Medical Records (often similar to people parallelism)
  - Protein or Gene **Sequences**;
  - Material properties, **Manufactured Object** specifications, etc., in custom dataset
  - Modelled entities like vehicles and people
- **Sensors** – Internet of Things
- **Events** such as detected anomalies in telescope or credit card data or atmosphere
- **(Complex) Nodes** in RDF Graph
- **Simple nodes** as in a learning network
- **Tweets, Blogs, Documents, Web Pages**, etc.
  - And characters/words in them
- **Files** or data to be backed up, moved or assigned metadata
- **Particles/cells/mesh points** as in parallel simulations
Features of 51 Use Cases I

- **PP (26)** “All” Pleasingly Parallel or Map Only
- **MR (18)** Classic MapReduce MR (add MRStat below for full count)
- **MRStat (7)** Simple version of MR where key computations are simple reduction as found in statistical averages such as histograms and averages
- **MRIter (23)** Iterative MapReduce or MPI (Spark, Twister)
- **Graph (9)** Complex graph data structure needed in analysis
- **Fusion (11)** Integrate diverse data to aid discovery/decision making; could involve sophisticated algorithms or could just be a portal
- **Streaming (41)** Some data comes in incrementally and is processed this way
- **Classify (30)** Classification: divide data into categories
- **S/Q (12)** Index, Search and Query
Features of 51 Use Cases II

- **CF (4)** Collaborative Filtering for recommender engines
- **LML (36)** Local Machine Learning (Independent for each parallel entity) – application could have GML as well
- **GML (23)** Global Machine Learning: Deep Learning, Clustering, LDA, PLSI, MDS,
  - Large Scale Optimizations as in Variational Bayes, MCMC, Lifted Belief Propagation, Stochastic Gradient Descent, L-BFGS, Levenberg-Marquardt. Can call EGO or Exascale Global Optimization with scalable parallel algorithm
- **Workflow (51)** Universal
- **GIS (16)** Geotagged data and often displayed in ESRI, Microsoft Virtual Earth, Google Earth, GeoServer etc.
- **HPC (5)** Classic large-scale simulation of cosmos, materials, etc. generating (visualization) data
- **Agent (2)** Simulations of models of data-defined macroscopic entities represented as agents
Local and Global Machine Learning

- Many applications use LML or Local machine Learning where machine learning (often from R) is run separately on every data item such as on every image.
- But others are GML Global Machine Learning where machine learning is a single algorithm run over all data items (over all nodes in computer).
  - maximum likelihood or $\chi^2$ with a sum over the N data items – documents, sequences, items to be sold, images etc. and often links (point-pairs).
  - Graph analytics is typically GML.
- Covering clustering/community detection, mixture models, topic determination, Multidimensional scaling, (Deep) Learning Networks.
- PageRank is “just” parallel linear algebra.
- Note many Mahout algorithms are sequential – partly as MapReduce limited; partly because parallelism unclear.
  - MLLib (Spark based) better.
- SVM and Hidden Markov Models do not use large scale parallelization in practice?
13 Image-based Use Cases

- **13-15 Military Sensor Data Analysis/ Intelligence**: PP, LML, GIS, MR
- **7: Pathology Imaging/ Digital Pathology**: PP, LML, MR for search becoming terabyte 3D images, Global Classification
- **18&35: Computational Bioimaging (Light Sources)**: PP, LML Also materials
- **26: Large-scale Deep Learning**: GML Stanford ran 10 million images and 11 billion parameters on a 64 GPU HPC; vision (drive car), speech, and Natural Language Processing
- **27: Organizing large-scale, unstructured collections of photos**: GML Fit position and camera direction to assemble 3D photo ensemble
- **36: Catalina Real-Time Transient Synoptic Sky Survey (CRTS)**: PP, LML followed by classification of events (GML)
- **43: Radar Data Analysis for CReSIS Remote Sensing of Ice Sheets**: PP, LML to identify glacier beds; GML for full ice-sheet
- **44: UAVSAR Data Processing, Data Product Delivery, and Data Services**: PP to find slippage from radar images
- **45, 46: Analysis of Simulation visualizations**: PP LML ?GML find paths, classify orbits, classify patterns that signal earthquakes, instabilities, climate, turbulence
Internet of Things and Streaming Apps

- It is projected that there will be 24 (Mobile Industry Group) to 50 (Cisco) billion devices on the Internet by 2020.
- The cloud natural controller of and resource provider for the Internet of Things.
- Smart phones/watches, Wearable devices (Smart People), “Intelligent River” “Smart Homes and Grid” and “Ubiquitous Cities”, Robotics.
- Majority of use cases are streaming – experimental science gathers data in a stream – sometimes batched as in a field trip. Below is sample
  - 10: Cargo Shipping Tracking as in UPS, Fedex PP GIS LML
  - 13: Large Scale Geospatial Analysis and Visualization PP GIS LML
  - 28: Truthy: Information diffusion research from Twitter Data PP MR for Search, GML for community determination
  - 39: Particle Physics: Analysis of LHC Large Hadron Collider Data: Discovery of Higgs particle PP for event Processing, Global statistics
  - 50: DOE-BER AmeriFlux and FLUXNET Networks PP GIS LML
  - 51: Consumption forecasting in Smart Grids PP GIS LML
Big Data Patterns – the Ogres
Benchmarking
Classifying Applications and Benchmarks

• “Benchmarks” “kernels” “algorithm” “mini-apps” can serve multiple purposes
• Motivate hardware and software features
  – e.g. collaborative filtering algorithm parallelizes well with MapReduce and suggests using Hadoop on a cloud
  – e.g. deep learning on images dominated by matrix operations; needs CUDA&MPI and suggests HPC cluster
• Benchmark sets designed cover key features of systems in terms of features and sizes of “important” applications
• Take 51 uses cases → derive specific features; each use case has multiple features
• Generalize and systematize as Ogres with features termed “facets”
• 50 Facets divided into 4 sets or views where each view has “similar” facets
7 Computational Giants of NRC Massive Data Analysis Report

http://www.nap.edu/catalog.php?record_id=18374

1) G1: Basic Statistics e.g. MRStat
2) G2: Generalized N-Body Problems
3) G3: Graph-Theoretic Computations
4) G4: Linear Algebraic Computations
5) G5: Optimizations e.g. Linear Programming
6) G6: Integration e.g. LDA and other GML
7) G7: Alignment Problems e.g. BLAST
HPC Benchmark Classics

- **Linpack** or HPL: Parallel LU factorization for solution of linear equations
- **NPB** version 1: Mainly classic HPC solver kernels
  - MG: Multigrid
  - CG: Conjugate Gradient
  - FT: Fast Fourier Transform
  - IS: Integer sort
  - EP: Embarrassingly Parallel
  - BT: Block Tridiagonal
  - SP: Scalar Pentadiagonal
  - LU: Lower-Upper symmetric Gauss Seidel
13 Berkeley Dwarfs

1) Dense Linear Algebra
2) Sparse Linear Algebra
3) Spectral Methods
4) N-Body Methods
5) Structured Grids
6) Unstructured Grids
7) MapReduce
8) Combinational Logic
9) Graph Traversal
10) Dynamic Programming
11) Backtrack and Branch-and-Bound
12) Graphical Models
13) Finite State Machines

First 6 of these correspond to Colella’s original.
Monte Carlo dropped.
N-body methods are a subset of Particle in Colella.

Note a little inconsistent in that MapReduce is a programming model and spectral method is a numerical method.
Need multiple facets!
Facets of the Ogres
Introducing Big Data Ogres and their Facets I

- **Big Data Ogres** provide a systematic approach to understanding applications, and as such they have **facets** which represent key characteristics defined both from our experience and from a bottom-up study of features from several individual applications.
- The facets capture common characteristics (shared by several problems) which are inevitably multi-dimensional and often overlapping.
- Ogres characteristics are cataloged in four distinct dimensions or views.
- Each view consists of facets; when multiple facets are linked together, they describe classes of big data problems represented as an Ogre.
- Instances of Ogres are particular big data problems.
- A set of Ogre instances that cover a rich set of facets could form a benchmark set.
- Ogres and their instances can be atomic or composite.
Introducing Big Data Ogres and their Facets II

- Ogres characteristics are cataloged in four distinct dimensions or views.
- Each view consists of facets; when multiple facets are linked together, they describe classes of big data problems represented as an Ogre.
- One view of an Ogre is the overall problem architecture which is naturally related to the machine architecture needed to support data intensive application while still being different.
- Then there is the execution (computational) features view, describing issues such as I/O versus compute rates, iterative nature 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.
- The final processing view has facets which describe classes of processing steps including algorithms and kernels. Ogres are specified by the particular value of a set of facets linked from the different views.
Facets of the Ogres

Problem Architecture

Meta or Macro Aspects of Ogres
Problem Architecture View of Ogres (Meta or MacroPatterns)

i. **Pleasingly Parallel** – as in BLAST, Protein docking, some (bio-)imagery including Local Analytics or Machine Learning – ML or filtering pleasingly parallel, as in bio-imagery, radar images (pleasingly parallel but sophisticated local analytics)

ii. **Classic MapReduce**: Search, Index and Query and Classification algorithms like collaborative filtering (G1 for MRStat in Features, G7)

iii. **Map-Collaborative**: Iterative maps + communication dominated by “collective” operations as in reduction, broadcast, gather, scatter. Common datamining pattern

iv. **Map-Point to Point**: Iterative maps + communication dominated by many small point to point messages as in graph algorithms

v. **Map-Streaming**: Describes streaming, steering and assimilation problems

vi. **Shared Memory**: Some problems are asynchronous and are easier to parallelize on shared rather than distributed memory – see some graph algorithms

vii. **SPMD**: Single Program Multiple Data, common parallel programming feature

viii. **BSP or Bulk Synchronous Processing**: well-defined compute-communication phases

ix. **Fusion**: Knowledge discovery often involves fusion of multiple methods.

x. **Dataflow**: Important application features often occurring in composite Ogres

xi. **Use Agents**: as in epidemiology (swarm approaches)

xii. **Workflow**: All applications often involve orchestration (workflow) of multiple components
Relation of Problem and Machine Architecture

- In my old papers (especially book Parallel Computing Works!), I discussed computing as multiple complex systems mapped into each other

Problem $\rightarrow$ Numerical formulation $\rightarrow$ Software $\rightarrow$ Hardware

- Each of these 4 systems has an architecture that can be described in similar language
- One gets an easy programming model if architecture of problem matches that of Software
- One gets good performance if architecture of hardware matches that of software and problem
- So “MapReduce” can be used as architecture of software (programming model) or “Numerical formulation of problem”
6 Forms of MapReduce cover “all” circumstances

Also an interesting software (architecture) discussion
Ogre Facets

Execution Features View
One View of Ogres has Facets that are micropatterns or **Execution Features**

i. **Performance Metrics**: property found by benchmarking Ogre

ii. **Flops per byte**: memory or I/O

iii. **Execution Environment; Core libraries needed**: matrix-matrix/vector algebra, conjugate gradient, reduction, broadcast; Cloud, HPC etc.

iv. **Volume**: property of an Ogre instance

v. **Velocity**: qualitative property of Ogre with value associated with instance

vi. **Variety**: important property especially of composite Ogres

vii. **Veracity**: important property of “mini-applications” but not kernels

viii. **Communication Structure**: Interconnect requirements; Is communication BSP, Asynchronous, Pub-Sub, Collective, Point to Point?

ix. Is application (graph) **static** or **dynamic**?

x. Most applications consist of a set of interconnected entities; is this **regular** as a set of pixels or is it a complicated **irregular graph**?

xi. Are algorithms **Iterative** or **not**?

xii. **Data Abstraction**: key-value, pixel, graph(G3), vector, bags of words or items

xiii. Are data points in **metric** or **non-metric spaces**?

xiv. Is algorithm **$O(N^2)$** or **$O(N)$** (up to logs) for N points per iteration (G2)
Facets of the Ogres

Data Source and Style Aspects
Data Source and Style View of Ogres I

i. **SQL NewSQL or NoSQL:** NoSQL includes Document, Column, Key-value, Graph, Triple store; NewSQL is SQL redone to exploit NoSQL performance

ii. **Other Enterprise data systems:** 10 examples from NIST integrate SQL/NoSQL

iii. **Set of Files or Objects:** as managed in iRODS and extremely common in scientific research

iv. **File systems, Object, Blob and Data-parallel** (HDFS) raw storage: Separated from computing or collocated? HDFS v Lustre v. Openstack Swift v. GPFS

v. **Archive/Batched/Streaming:** Streaming is incremental update of datasets with new algorithms to achieve real-time response (G7); Before data gets to compute system, there is often an initial data gathering phase which is characterized by a block size and timing. Block size varies from month (Remote Sensing, Seismic) to day (genomic) to seconds or lower (Real time control, streaming)
vi. **Shared/Dedicated/Transient/Permanent**: qualitative property of data; Other characteristics are needed for permanent auxiliary/comparison datasets and these could be interdisciplinary, implying nontrivial data movement/replication.

vii. **Metadata/Provenance**: Clear qualitative property but not for kernels as important aspect of data collection process.

viii. **Internet of Things**: 24 to 50 Billion devices on Internet by 2020.

ix. **HPC simulations**: generate major (visualization) output that often needs to be mined.

x. **Using GIS**: Geographical Information Systems provide attractive access to geospatial data.

Note 10 Bob Marcus (led NIST effort) Use cases
2. Perform real time analytics on data source streams and notify users when specified events occur.

- Streaming Data
- Streaming Data
- Streaming Data

Fetch streamed data → Posted Data

Filter identifying events → Identified Events

Specify filter → Filter Identifying Events

Archive → Repository

Post Selected Events → Identified Events

Storm, Kafka, Hbase, Zookeeper
5. Perform interactive analytics on data in analytics-optimized database

Data Storage: HDFS, Hbase

Hadoop, Spark, Giraph, Pig ...

Mahout, R

Data, Streaming, Batch .....
5A. Perform interactive analytics on observational scientific data

Science Analysis Code, Mahout, R

Grid or Many Task Software, Hadoop, Spark, Giraph, Pig ...

Data Storage: HDFS, Hbase, File Collection

Direct Transfer

Streaming Twitter data for Social Networking

Record Scientific Data in “field”

Local Accumulate and initial computing

Transport batch of data to primary analysis data system

NIST examples include LHC, Remote Sensing, Astronomy and Bioinformatics
Facets of the Ogres

Processing View
Facets in **Processing** (runtime) View of Ogres I

i. **Micro-benchmarks** ogres that exercise simple features of hardware such as communication, disk I/O, CPU, memory performance

ii. **Local Analytics** executed on a single core or perhaps node

iii. **Global Analytics** requiring iterative programming models (G5,G6) across multiple nodes of a parallel system

iv. **Optimization Methodology**: overlapping categories
   i. **Nonlinear Optimization** (G6)
   ii. **Machine Learning**
   iii. **Maximum Likelihood** or $\chi^2$ minimizations
   iv. **Expectation Maximization** (often Steepest descent)
   v. **Combinatorial Optimization**
   vi. **Linear/Quadratic Programming** (G5)
   vii. **Dynamic Programming**

v. **Visualization** is key application capability with algorithms like MDS useful but it itself part of “mini-app” or composite Ogre

vi. **Alignment** (G7) as in BLAST compares samples with repository
Facets in Processing (run time) View of Ogres II

vii. Streaming divided into 5 categories depending on event size and synchronization and integration
- Set of independent events where precise time sequencing unimportant.
- Time series of connected small events where time ordering important.
- Set of independent large events where each event needs parallel processing with time sequencing not critical.
- Set of connected large events where each event needs parallel processing with time sequencing critical.
- Stream of connected small or large events to be integrated in a complex way.

viii. Basic Statistics (G1): MRStat in NIST problem features

ix. Search/Query/Index: Classic database which is well studied (Baru, Rabl tutorial)

x. Recommender Engine: core to many e-commerce, media businesses; collaborative filtering key technology

xi. Classification: assigning items to categories based on many methods
- MapReduce good in Alignment, Basic statistics, S/Q/I, Recommender, Classification

xii. Deep Learning of growing importance due to success in speech recognition etc.

xiii. Problem set up as a graph (G3) as opposed to vector, grid, bag of words etc.

xiv. Using Linear Algebra Kernels: much machine learning uses linear algebra kernels
Benchmarks and Ogres
Benchmarks/Mini-apps spanning Facets

- **Look at** NSF SPIDAL 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
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Applications</th>
<th>Problem Arch View</th>
<th>Execution View</th>
<th>Processing View</th>
</tr>
</thead>
<tbody>
<tr>
<td>DA Vector Clustering</td>
<td>Accurate Clusters</td>
<td>3, 7, 8</td>
<td>9D, 10I, 11, 12V, 13M, 14N</td>
<td>9ML, 9EM, 12</td>
</tr>
<tr>
<td>DA Non-metric Clustering</td>
<td>Accurate Clusters, Biology, Web</td>
<td>3, 7, 8</td>
<td>9S, 10R, 11, 12V, 13M, 14NN</td>
<td>9ML, 9EM, 12</td>
</tr>
<tr>
<td>Kmeans; Basic, Fuzzy and Elkan</td>
<td>Fast Clustering</td>
<td>3, 7, 8</td>
<td>9D, 10I(Elkan), 11, 12V, 13M, 14N</td>
<td>9ML, 9EM</td>
</tr>
<tr>
<td>Levenberg-Marquardt Optimization</td>
<td>Non-linear Gauss-Newton, use in MDS</td>
<td>3, 7, 8</td>
<td>9D, 10R, 11, 12V, 14NN</td>
<td>9ML, 9NO, 9LS, 9EM, 12</td>
</tr>
<tr>
<td>DA, Weighted SMACOF</td>
<td>MDS with general weights</td>
<td>3, 7, 8</td>
<td>9S, 10R, 11, 12BI, 13N, 14NN</td>
<td>9ML, 9NO, 9LS, 9EM, 12, 14</td>
</tr>
<tr>
<td>TFIDF Search</td>
<td>Find nearest neighbors in document corpus</td>
<td>1</td>
<td>9S, 10R, 12BI, 13N, 14N</td>
<td>2, 9ML</td>
</tr>
<tr>
<td>All-pairs similarity search</td>
<td>Find pairs of documents with TFIDF distance below a threshold</td>
<td>3, 7, 8</td>
<td>9S, 10R, 12BI, 13N, 14NN</td>
<td>9ML</td>
</tr>
<tr>
<td>Support Vector Machine SVM</td>
<td>Learn and Classify</td>
<td>3, 7, 8</td>
<td>9S, 10R, 11, 12V, 13M, 14N</td>
<td>7, 8, 9ML</td>
</tr>
<tr>
<td>Random Forest</td>
<td>Learn and Classify</td>
<td>1</td>
<td>9S, 10R, 12BI, 13N, 14N</td>
<td>2, 7, 8, 9ML</td>
</tr>
<tr>
<td>Gibbs sampling (MCMC)</td>
<td>Solve global inference problems</td>
<td>3, 7, 8</td>
<td>9S, 10R, 11, 12BW, 13N, 14N</td>
<td>9ML, 9NO, 9EM</td>
</tr>
<tr>
<td>Latent Dirichlet Allocation LDA with Gibbs sampling or Var. Bayes</td>
<td>Topic models (Latent factors)</td>
<td>3, 7, 8</td>
<td>9S, 10R, 11, 12BW, 13N, 14N</td>
<td>9ML, 9EM</td>
</tr>
<tr>
<td>Singular Value Decomposition SVD</td>
<td>Dimension Reduction and PCA</td>
<td>3, 7, 8</td>
<td>9S, 10R, 11, 12V, 13M, 14NN</td>
<td>9ML, 12</td>
</tr>
<tr>
<td>Hidden Markov Models (HMM)</td>
<td>Global inference on sequence models</td>
<td>3, 7, 8</td>
<td>9S, 10R, 11, 12BI</td>
<td>2, 9ML, 12</td>
</tr>
<tr>
<td>Facet and View</td>
<td>Comments</td>
<td>SP</td>
<td>DB</td>
<td>NI</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td><strong>Facets in Problem Architecture View (AV)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Pleasingly Parallel</td>
<td>Clear qualitative property overlapping Local Analytics</td>
<td>M</td>
<td>S</td>
<td>H</td>
</tr>
<tr>
<td>2. Classic MapReduce</td>
<td>Clear qualitative property of non-iterative algorithms</td>
<td>M</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>4. Map Point-to-Point (graphs)</td>
<td>Clear qualitative property of graphs and simulation</td>
<td>H</td>
<td>S</td>
<td>M</td>
</tr>
<tr>
<td>5. Map Streaming</td>
<td>Property of growing importance. Not well benchmarked</td>
<td>N</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>6. Shared memory (as opposed to distributed parallel algorithm)</td>
<td>Corresponds to problem where shared memory implementations important. Tend to be dynamic asynchronous</td>
<td>S</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>7. Single Program Multiple Data SPMD</td>
<td>Clear qualitative property famous in parallel computing</td>
<td>H</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>8. Bulk Synchronous Processing BSP</td>
<td>Needs to be defined but reasonable qualitative property</td>
<td>H</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>9. Fusion</td>
<td>Only present for composite Ogres</td>
<td>N</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>10. Dataflow</td>
<td>Only present for composite Ogres</td>
<td>N</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>11. Agents</td>
<td>Clear but uncommon qualitative property</td>
<td>N</td>
<td>N</td>
<td>S</td>
</tr>
<tr>
<td>12. Orchestration (workflow)</td>
<td>Only present for composite Ogres</td>
<td>N</td>
<td>H</td>
<td>H</td>
</tr>
</tbody>
</table>
## Facets in Execution View (EV)

<table>
<thead>
<tr>
<th>Facet and View</th>
<th>Comments</th>
<th>SP</th>
<th>DB</th>
<th>NI</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. Performance Metrics</strong></td>
<td>Result of Benchmark</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>2. Flops per Byte (Memory or I/O). Flops per watt (power).</strong></td>
<td>I/O Not needed for “pure in memory” benchmark. Value needs detailed quantitative study. Could depend on implementation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>3. Execution Environment (LN = Libraries needed, C= Cloud, HPC = HPC, T=Threads, MP= Message Passing)</strong></td>
<td>Depends on how benchmark set up. Could include details of machine used for benchmarking here</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>4. Volume</strong></td>
<td>Depends on data size. Benchmark measure</td>
<td></td>
<td>M</td>
<td></td>
</tr>
<tr>
<td><strong>5. Velocity</strong></td>
<td>Associated with streaming facet but value depends on particular problem</td>
<td>N</td>
<td>S</td>
<td>H</td>
</tr>
<tr>
<td><strong>6. Variety</strong></td>
<td>Most useful for composite Ogres</td>
<td>N</td>
<td>S</td>
<td>H</td>
</tr>
<tr>
<td><strong>7. Veracity</strong></td>
<td>Most problems would not discuss but potentially important</td>
<td>N</td>
<td>N</td>
<td>M</td>
</tr>
<tr>
<td><strong>8. Communication Structure (D=Distributed, I=Interconnect, S=Synchronization)</strong></td>
<td>Qualitative property – related to BSP (Bulk Synchronous Processing) and Shared memory</td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td><strong>9. D=Dynamic or S=Static</strong></td>
<td>Clear qualitative properties. Importance familiar from parallel computing</td>
<td>H</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td><strong>10. R=Regular or I=Irregular</strong></td>
<td>Clear qualitative property. Highlighted by Iterative MapReduce and always present in classic parallel computing</td>
<td>H</td>
<td>S</td>
<td>H</td>
</tr>
<tr>
<td><strong>11. Iterative?</strong></td>
<td>Clear qualitative property although important data abstractions not agreed upon. All should be supported by Programming model and run time</td>
<td>H</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td><strong>12. Data Abstraction(K= key-value, BW= bag of words, BI = bag of items, P= pixel/spatial, V= vectors/matrices, S= sequence, G= graph)</strong></td>
<td>Clear qualitative property discussed in [69]</td>
<td>H</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>Facet and View</td>
<td>Comments</td>
<td>SP</td>
<td>DB</td>
<td>NI</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>1 SQL/NoSQL/NewSQL?</td>
<td>Clear qualitative property. Can add NoSQL sub-categories such as key-value, graph, document …</td>
<td>N</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>2 Enterprise data model (warehouses)</td>
<td>Clear qualitative property of data model highlighted in database community / industry benchmarks</td>
<td>N</td>
<td>H</td>
<td>M</td>
</tr>
<tr>
<td>3 Files/Objects?</td>
<td>Clear qualitative property of data model where files important in Science; objects in industry</td>
<td>N</td>
<td>S</td>
<td>H</td>
</tr>
<tr>
<td>4 HDFS/Lustre/GPFS?</td>
<td>Clear qualitative property where HDFS important in Apache stack but not much used in science</td>
<td>N</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>5 Archive/Batched/Streaming</td>
<td>Clear qualitative property but not for kernels as it describes how data is collected</td>
<td>N</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>6 Shared/Dedicated/Transient/Permanent</td>
<td>Clear qualitative property of data whose importance is not well studied</td>
<td>N</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>7 Metadata/Provenance</td>
<td>Clear qualitative property but not for kernels as important aspect of data collection process</td>
<td>N</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>8 Internet of Things</td>
<td>Dominant source of commodity data in future</td>
<td>N</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>9 HPC Simulations</td>
<td>Important in science research especially at exascale</td>
<td>N</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>10 Geographic Information Systems</td>
<td>Clear property but not for kernels</td>
<td>S</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>Facet and View</td>
<td>Comments</td>
<td>SP</td>
<td>DB</td>
<td>NI</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>1 Micro-benchmarks</td>
<td>Important subset of small kernels</td>
<td>N</td>
<td>H</td>
<td>N</td>
</tr>
<tr>
<td>2 Local Analytics or Informatics</td>
<td>Well defined but overlaps Pleasingly Parallel</td>
<td>H</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>3 Global Analytics or Informatics</td>
<td>Clear qualitative property that includes parallel Mahout (E.g. Kmeans) and Hive (database)</td>
<td>H</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>4 Base Statistics</td>
<td>Describes simple statistical averages needing simple MapReduce. MRStat in [6]</td>
<td>N</td>
<td>N</td>
<td>M</td>
</tr>
<tr>
<td>5 Recommender Engine</td>
<td>Clear type of machine learning of especial importance commercially</td>
<td>N</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>6 Search/Query/Index</td>
<td>Clear important class of algorithms in industry</td>
<td>S</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>7 Classification</td>
<td>Clear important class of algorithms</td>
<td>S</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>8 Learning</td>
<td>Includes deep learning as category</td>
<td>S</td>
<td>S</td>
<td>H</td>
</tr>
<tr>
<td>9 Optimization Methodology (ML= Machine Learning, NO = Nonlinear Optimization, LS = Least Squares, EM = expectation maximization, LQP = Linear/Quadratic Programming, CO = Combinatorial Optimization)</td>
<td>LQP and CO overshadowed by machine learning but important where used. ML includes many analytics which are often NO and EM and sometimes LS (or similar Maximum Likelihood)</td>
<td>H</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>10 Streaming</td>
<td>Clear important class of algorithms associated with Internet of Things. Can be called DDDAS Dynamic Data-Driven Application Systems</td>
<td>N</td>
<td>N</td>
<td>H</td>
</tr>
<tr>
<td>11 Alignment</td>
<td>Clear important class of algorithms as in BLAST</td>
<td>N</td>
<td>S</td>
<td>M</td>
</tr>
<tr>
<td>12 Linear Algebra Kernels</td>
<td>Important property of some analytics</td>
<td>H</td>
<td>S</td>
<td>H</td>
</tr>
<tr>
<td>13 Graph Algorithms</td>
<td>Clear important class of algorithms – often hard</td>
<td>H</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>14 Visualization</td>
<td>Clearly important aspect of data analysis but different in character to most other facets</td>
<td>S</td>
<td>N</td>
<td>H</td>
</tr>
</tbody>
</table>
Conclusions

- Collected 51 use cases; useful although certainly incomplete and biased (to research and against energy for example)
- Improved (especially in security and privacy) and available as online form
- Identified 50 features called facets divided into 4 sets (views) used to classify applications
- Used to derive set of hardware architectures
  - Could discuss software (see papers)
- Surveyed some benchmarks
- Could be used to identify missing benchmarks
  - Noted streaming a dominant feature of use cases but not common in benchmarks
Spare Slides
8 Data Analysis Problem Architectures

1) Pleasingly Parallel PP or “map-only” in MapReduce
   - BLAST Analysis; Local Machine Learning

2A) Classic MapReduce MR, Map followed by reduction
   - High Energy Physics (HEP) Histograms; Web search; Recommender Engines

2B) Simple version of classic MapReduce MRStat
   - Final reduction is just simple statistics

3) Iterative MapReduce MRIter
   - Expectation maximization Clustering Linear Algebra, PageRank

4A) Map Point to Point Communication
   - Classic MPI; PDE Solvers and Particle Dynamics; Graph processing Graph

4B) GPU (Accelerator) enhanced 4A) – especially for deep learning

5) Map + Streaming + Communication
   - Images from Synchrotron sources; Telescopes; Internet of Things IoT

6) Shared memory allowing parallel threads which are tricky to program but lower latency
   - Difficult to parallelize asynchronous parallel Graph Algorithms
## Kaleidoscope of (Apache) Big Data Stack (ABDS) and HPC Technologies

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>17) Workflow-Orchestration</td>
<td>ODE, ActiveBPEL, Airavata, Pegasus, Kepler, Swift, Taverna, Triana, Trident, BioKepler, Galaxy, IPython, Dryad, Naiad, Oozie, Tez, Google FlumeJava, Crunch, Cascading, Scalding, e-Science Central, Azure Data Factory, Google Cloud Dataflow, NiFi (NSA), Jitterbit, Talend, Pentaho, Apator</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16) Application and Analytics</td>
<td>Mahout, MLib, MBase, DataFu, R, pbdR, Bioconductor, ImageJ, OpenCV, Scalapack, PetSc, Azure Machine Learning, Google Prediction API &amp; Translation API, mlpy, scikit-learn, PyBrain, CompLearn, DAAL (Intel), Caffe, Torch, Theano, DL4j, H2O, IBM Watson, Oracle PGX, GraphLab, GraphX, MapGraph, IBM System G, GraphBuilder (Intel), TinkerPop, Google Fusion Tables, CINET, NWB, Elasticsearch, Kibana, Logstash, Graylog, Splunk, Tableau, D3.js, three.js, Potree</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15A) High level Programming</td>
<td>Kite, Hive, HCatalog, Tajo, Shark, Phoenix, Impala, MRQL, SAP HANA, HadoopDB, PolyBase, Pivotal HD/Hawq, Presto, Google Dremel, Google BigQuery, Amazon Redshift, Drill, Kyoto Cabinet, Pig, Sawzall, Google Cloud DataFlow, Summingbird</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14B) Streams</td>
<td>Storm, S4, Samza, Granules, Google MillWheel, Amazon Kinesis, LinkedIn Databus, Facebook Puma/Ptlai/Scribe/ODS, Azure Stream Analytics</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14A) Basic Programming model and runtime, SPMD, MapReduce</td>
<td>Hadoop, Spark, Twister, Stratosphere (Apache Flink), Reef, Hama, Giraph, Pregel, Pegasys, Ligra, GraphChi</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13) Inter process communication Collectives, point-to-point, publish-subscribe</td>
<td>MPI, Harp, Netty, ZeroMQ, ActiveMQ, RabbitMQ, NaradaBrokering, QPid, Kafka, Kestrel, JMS, AMQP, Stomp, MQTT, Public Cloud: Amazon SNS, Lambda, Google Pub Sub, Azure Queues, Event Hubs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12) In-memory databases/caches</td>
<td>Gora (general object from NoSQL), Memcached, Redis, LMDB (key value), Hazelcast, Ehcache, Infinispan</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12) Object-relational mapping</td>
<td>Hibernate, OpenJPA, EclipseLink, DataNucleus, ODBC/JDBC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12) Extraction Tools</td>
<td>UIMA, Tika</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11C) SQL(NewSQL): Oracle, DB2, SQL Server, SQLite, MySQL, PostgreSQL, CUBRID, Galera Cluster, SciDB, Rasdaman, Apache Derby, Pivotal Greenplum, Google Cloud SQL, Azure SQL, Amazon RDS, Google F1, IBM dashDB, N1QL, BlinkDB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11B) NoSQL: Lucene, Solr, Solandra, Voldemort, Riak, Berkeley DB, Kyoto/Tokyo Cabinet, Tycoon, Yrant, MongoDB, Expresso, CouchDB, Couchbase, IBM Cloudant, Pivotal Gemfire, HBase, Google Bigtable, LevelDB, Megastore and Spanner, Accumulo, Cassandra, RYA, Sqrrl, Neo4J, Yardata, AllegroGraph, Blazegraph, Facebook Tao, Titan:db, Jena, Sesame</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11A) File management</td>
<td>iRODS, NetCDF, CDF, HDF, OPeNDAP, FITS, RCFie, ORC, Parquet</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10) Data Transport</td>
<td>BitTorrent, HTTP, FTP, SSH, Globus Online (GridFTP), Flume, Sqoop, Pivotal GLOAD/GPFDIST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9) Cluster Resource Management</td>
<td>Mesos, Yarn, Helix, Lima, Google Omega, Facebook Corona, Celery, HTCondor, SGE, OpenPBS, Moab, Slurm, Torque, Globus Tools, Pilot Jobs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8) File systems</td>
<td>HDFS, Swift, Haystack, f4, Cinder, Ceph, FUSE, Gluster, Lustre, GPFS, GFFS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Public Cloud</td>
<td>Amazon S3, Azure Blob, Google Cloud Storage</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7) Interoperability</td>
<td>Libvirt, Libcloud, JClouds, TOCSA, OCCI, CDMI, Whirr, Saga, Genesis</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5) IaaS Management from HPC to hypervisors</td>
<td>Xen, KVM, Hyper-V, VirtualBox, OpenVZ, LXC, Linux-Vserver, OpenStack, OpenNebula, Eucalyptus, Nimbus, CloudStack, CoreOS, VMware ESXi, vSphere and vCloud, Amazon, Azure, Google and other public Clouds, Networking: Google Cloud DNS, Amazon Route 53</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Functionality of 21 HPC-ABDS Layers

1) Message Protocols:
2) Distributed Coordination:
3) Security & Privacy:
4) Monitoring:
5) IaaS Management from HPC to hypervisors:
6) DevOps:
7) Interoperability:
8) File systems:
9) Cluster Resource Management:
10) Data Transport:
11) A) File management
    B) NoSQL
    C) SQL
12) In-memory databases&caches / Object-relational mapping / Extraction Tools
13) Inter process communication Collectives, point-to-point, publish-subscribe, MPI:
14) A) Basic Programming model and runtime, SPMD, MapReduce:
    B) Streaming:
15) A) High level Programming:
    B) Frameworks
16) Application and Analytics:
17) Workflow-Orchestration:

Here are 21 functionalities.
(including 11, 14, 15 subparts)

4 Cross cutting at top
17 in order of layered diagram starting at bottom
Software for a Big Data Initiative

- Functionality of ABDS and Performance of HPC
- **Workflow:** Apache Crunch, Python or Kepler
- **Data Analytics:** Mahout, R, ImageJ, Scalapack
- **High level Programming:** Hive, Pig
- **Batch Parallel Programming model:** Hadoop, Spark, Giraph, Harp, MPI;
- **Streaming Programming model:** Storm, Kafka or RabbitMQ
- **In-memory:** Memcached
- **Data Management:** Hbase, MongoDB, MySQL
- **Distributed Coordination:** Zookeeper
- **Cluster Management:** Yarn, Slurm
- **File Systems:** HDFS, Object store (Swift), Lustre
- **DevOps:** Cloudmesh, Chef, Puppet, Docker, Cobbler
- **IaaS:** Amazon, Azure, OpenStack, Docker, SR-IOV
- **Monitoring:** Inca, Ganglia, Nagios
Pleasingly Parallel

Classic MapReduce

Orchestration (Workflow)

Fusion

Bulk Synchronous Processing BSP

Single Program Multiple Data

SPMD

Map-Collective

Map Point-to-Point

Map-Streaming

Agents

Shared Memory

Dataflow?

Ogres Problem Architecture View