Towards a Comprehensive Set of Big Data Benchmarks

Geoffrey C. FOX [a, 1], Shantenu JHA[b], Judy QIU[a], Saliya EKANAYAKE[a] and Andre LUCKOW[b]
[a] *School of Informatics and Computing, Indiana University, Bloomington, IN 47408, USA*
[b] *RADICAL, Rutgers University, Piscataway, NJ 08854, USA*

Abstract. This paper reviews the Ogre classification of Big Data application with 50 facets divided into four groups or views. These four correspond to Problem Architecture, Execution mode, Data source and style, and the Processing model used. We then look at multiple existing or proposed benchmark suites and analyze their coverage of the different facets suggesting a process to obtain a complete set. We illustrate this by looking at parallel data analytics benchmarked on multicore clusters.

Keywords. Big Data, Benchmarking, Analytics, Database

1. Introduction

We propose a systematic approach to Big Data benchmarking based on a recent classification (Ogres) of applications using a set of facets or features divided into 4 dimensions: Problem Architecture (or Structure), Execution mode, Data source, storage and access, and the Processing algorithms used. This is reviewed in Section 2 and summarized in a detailed Table given in the Appendix. We analyze many existing and proposed benchmark sets in section 3 and show how they cover the set of facets. We give some examples of benchmarking data analytics on clusters in Section 4 and propose further steps in Section 5.

2. Overview of Ogres

2.1. What is an Ogre?

The Berkeley Dwarfs [2] were an important step towards defining an exemplar set of parallel (high performance computing) applications. The recent NRC report [3] gave Seven Computational Giants of Massive Data Analysis, which start to define critical types of data analytics problems. We propose Ogres [4-6] — an extension of these ideas based on an analysis by NIST of 51 big data applications [7, 8]. 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

[1] Corresponding Author.

study of features from several individual applications. The facets capture common characteristics which are inevitably multi-dimensional and often overlapping. We note that in HPC, the Berkeley Dwarfs were very successful as patterns but did not get adopted as a standard benchmark set. Rather the NAS Parallel Benchmarks [9], Linpack [10], and (mini-)applications played this role. This suggests that benchmarks do not follow directly from patterns, but the latter can help by allowing one to understand breadth of applications covered by a benchmark set.

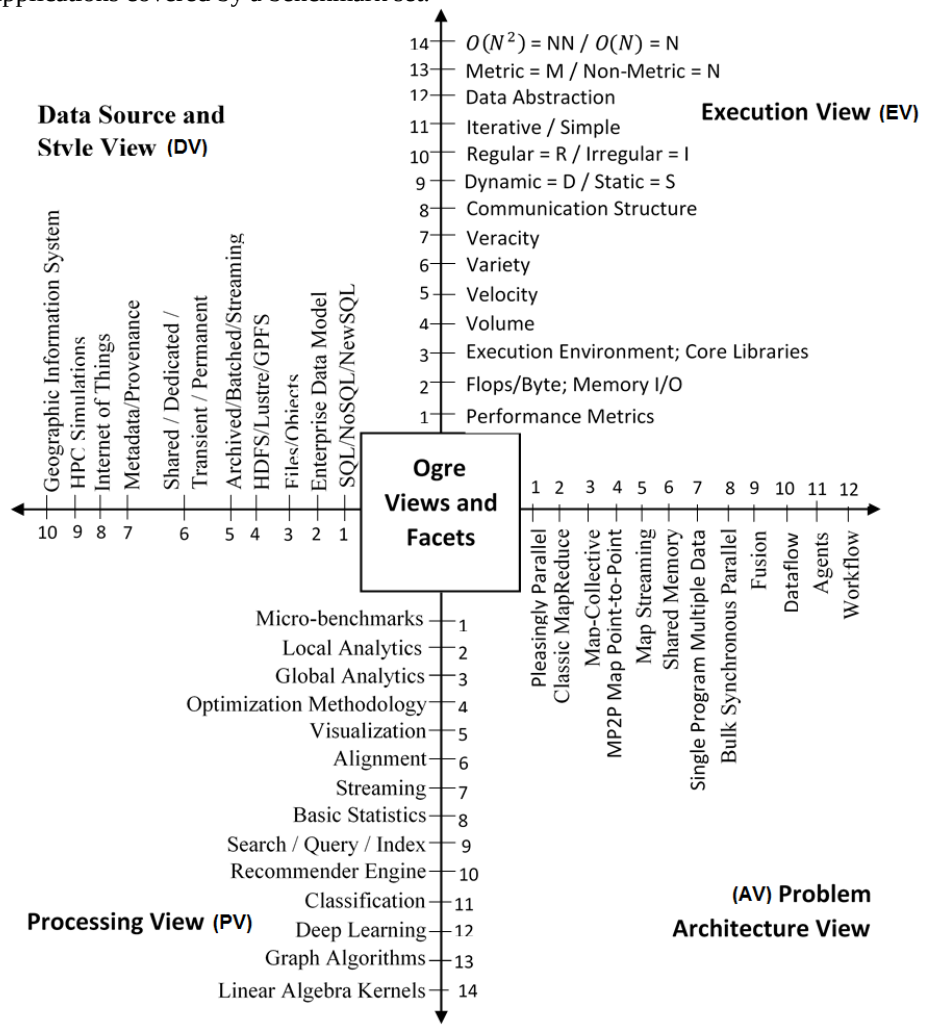

Figure 1: The 4 Ogre Views and their Facets

2.2. Ogres have Facets

We suggest Ogres possess properties that we classify 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** (labelled **AV**) which is naturally related to the machine architecture needed to support data intensive application. Then there is the **execution (computational) features** view (labelled **EV**), 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 (labelled **DV**) includes facets specifying how the data is collected, stored and accessed. The final **processing** view (labelled **PV**) 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.

The views are illustrated in Figure 1 and listed in Table A in the Appendix. Table A also lists in its last 3 columns a measure of the coverage of two types of benchmarks – those of SPIDAL from Section 3.1 and those of the database survey given in section 3.2 with the rightmost column listing the application coverage of the NIST survey [7, 8].

In our language, instances of Ogres can form benchmarks. One can consider composite or atomic (simple, basic) benchmarks. For example, a clustering benchmark is an instance of an Ogre with a Map-Collective facet in the Problem Architecture view and the machine learning sub-facet in the Processing view. The Execution view describes properties that could vary for different clustering algorithms and would often be measured in a benchmarking process. Note a simple benchmark like this could ignore the Data Source & Style view and just be studied for in-memory data. Alternatively we can consider a composite benchmark linking clustering to different data storage mechanisms. A given benchmark can be associated with multiple facets in a single view, i.e. clustering has other problem architecture facets including SPMD, BSP, and Global Analytics.

3. Particular Benchmarks as instances of Ogres

Our approach suggests choosing benchmarks from Ogre instances that cover a diverse range of facets. Rather than trying to be comprehensive at this stage, we give some examples. Note that kernel benchmarks are instances of Ogre Processing facets classified as PV2 to PV13; this is where the NAS parallel benchmarks or TeraSort [11] would fit. Further we have micro-benchmarks such as MPI ping-pong and SPEC [12] as facet PV1 and giving measures of Ogre execution facets EV1 to EV5. In sections 3.1, 3.2 and 3.3 we go through 3 different sources of benchmarks comparing them to facets.

3.1. SPIDAL Library as Ogre Benchmarks

We are part of a recently started NSF project from the DIBBs (Data Infrastructure Building Blocks) program where one can use Ogres to classify Building Blocks that are part of the SPIDAL Scalable Parallel Interoperable Data Analytics Library, which is the focus of this program. In Table 1, we list the proposed library members from this project with more details available [13, 14]. Note each problem can provide benchmarks for many different execution view facets. For example the first set (Graph Analytics) are all instances of the Graph Algorithm facet of the Processing view and either the Map Point-to-Point and/or Shared memory facets in the Problem architecture view. In the second group of spatial kernels, we find *queries* from the Search/Query/Index and MapReduce facets with *Spatial Clustering* from Global Machine Learning, Map-Collective and Global Analytics facets and *Distance-based queries* from Pleasingly Parallel and Search/Query/Index facets. These benchmarks all have the spatial data abstraction facet

and could be involved with GIS facet. In Machine Learning in general and for image processing categories we find several *Clustering* algorithms illustrating O(N), O(N^2), and Metric (non-metric) space execution view facets; *Levenberg-Marquardt Optimization* and *SMACOF Multi-Dimensional Scaling* with Linear Algebra Kernels and Expectation maximization/least squares characteristic in optimization methodology facet from Processing view; *TFIDF Search and Random Forest* with Pleasingly Parallel facets. All exhibit the machine learning sub-facet of the processing view. In the last 3 columns of Table 1, we quantify this, listing facets for each SPIDAL library member for three of the four facet views. The analytics focus of this project implies little overlap with the Data Source & Style view and some of the entries are preliminary estimates which need more study. Further the SPIDAL project reviewed the Apache libraries MLlib and Mahout in choosing their library members.

	Algorithm	Applications	Problem Arch View	Execution View	Processing View
	Graph Analytics GA				
1	**Community detection**	Social networks, webgraph	3, 4, 7	9S, 10I, 11, 12G	3, 9ML, 13
2	**Subgraph/motif finding**	Webgraph, biological/social networks	4, 7	9D, 10I, 12G	3, 9ML, 13
3	**Finding diameter**	Social networks, webgraph	4, 7	9D, 10I, 12G	3, 9ML, 13
4	**Clustering coefficient**	Social networks	4, 7	9S, 10I, 11, 12G	3, 9ML, 13
5	**Page rank**	Webgraph	3, 4, 7	9S, 10I, 11, 12V	3, 9ML, 12, 13
6	**Maximal cliques**	Social networks, webgraph	4, 7	9D, 10I, 12G	3, 9ML, 13
7	**Connected component**	Social networks, webgraph	4, 7	9D, 10I, 12G	3, 9ML, 13
8	**Betweenness centrality**	Social networks	6	9D, 10I, 12G, 13N	9ML, 13
9	**Shortest path**	Social networks, webgraph	6	9D, 10I, 12G, 13N	9ML, 13
	Spatial Queries and Analytics SQA				
1	**Spatial relationship based queries**	GIS/social networks/pathology informatics (add GIS in data view)	2	12P	6
2	**Distance based queries**		1	12P	2
3	**Spatial clustering**		3, 7, 8	12P	3, 9ML,EM
4	**Spatial modeling**		1	12P	2
	Core Image Processing IP				
1	**Image preprocessing**	Computer vision/pathology informatics	1	13M	2
2	**Object detection & segmentation**		1	13M	2, 9ML
3	**Image/object feature computation**		1	13M	2, 9ML
4	**3D image registration**		1	13M	2, 9ML
5	**Object matching**		1	13N	2, 9ML
6	**3D feature extraction**		1	13N	2, 9ML
	General Machine Learning GML				
1	**DA Vector Clustering**	Accurate Clusters	3, 7, 8	9D, 10I, 11, 12V, 13M, 14N	9ML, 9EM, 12

2	**DA Non-metric Clustering**	Accurate Clusters, Biology, Web	3, 7, 8	9S, 10R, 11, 12BI, 13N, 14NN	9ML, 9EM, 12
3	**Kmeans; Basic, Fuzzy and Elkan**	Fast Clustering	3, 7, 8	9D, 10I(Elkan), 11, 12V, 13M, 14N	9ML, 9EM
4	**Levenberg-Marquardt Optimization**	Non-linear Gauss-Newton, use in MDS	3, 7, 8	9D, 10R, 11, 12V, 14NN	9ML, 9NO, 9LS, 9EM, 12
5	**DA, Weighted SMACOF**	MDS with general weights	3, 7, 8	9S, 10R, 11, 12BI, 13N, 14NN	9ML, 9NO, 9LS, 9EM, 12, 14
6	**TFIDF Search**	Find nearest neighbors in document corpus	1	9S, 10R, 12BI, 13N, 14N	2, 9ML
7	**All-pairs similarity search**	Find pairs of documents with TFIDF distance below a threshold	3, 7, 8	9S, 10R, 12BI, 13N, 14NN	9ML
8	**Support Vector Machine SVM**	Learn and Classify	3, 7, 8	9S, 10R, 11, 12V, 13M, 14N	7, 8, 9ML
9	**Random Forest**	Learn and Classify	1	9S, 10R, 12BI, 13N, 14N	2, 7, 8, 9ML
10	**Gibbs sampling (MCMC)**	Solve global inference problems	3, 7, 8	9S, 10R, 11, 12BW, 13N, 14N	9ML, 9NO, 9EM
11	**Latent Dirichlet Allocation LDA with Gibbs sampling or Var. Bayes**	Topic models (Latent factors)	3, 7, 8	9S, 10R, 11, 12BW, 13N, 14N	9ML, 9EM
12	**Singular Value Decomposition SVD**	Dimension Reduction and PCA	3, 7, 8	9S, 10R, 11, 12V, 13M, 14NN	9ML, 12
13	**Hidden Markov Models (HMM)**	Global inference on sequence models	3, 7, 8	9S, 10R, 11, 12BI	2, 9ML, 12

Table 1: The proposed members of SPIDAL library [13] and the Ogre facets that they support. There are no SPIDAL library members directly addressing Data Source & Style View (except spatial analytics and GIS) and so that is omitted.

3.2. Well Established Data Systems and Database Benchmarks

Big Data has an excellent base set of benchmarks coming from the long established efforts of the database community with important Industry contributions. We build on Baru and Rabl's excellent tutorial [15], which has a thorough discussion of benchmarks including the TPC series [16], HiBench [17], Yahoo Cloud Serving Benchmark [18], BigDataBench [19], BigBench [20] and Berkeley Big Data Benchmark [21] that quantify the Ogre Data Source & Style facets. We summarize these and other important benchmarks from Europe [22], SPEC [23] and other micro-benchmark [24, 25] and analytics and social networking studies [26-29] in the following sections.

3.2.1. Micro Benchmarks

The SPEC Benchmarks [23] are well known here and they cover a variety of areas including CPU, HPC, Servers, Power, Virtualization and the Web. SPEC has set up a

Big Data working group [30] that will further improve SPEC benchmarks in this area. There are several studies of I/O performance including use of flash storage [24] and HDFS [25]. These types of benchmarks correspond to Ogre facets that include DV3-4 and PV1-2.

3.2.2. Enterprise Database Benchmarks: TPC

The Transaction Processing Performance Council TPC [16] benchmarks are well-known and central to the database industry. TPC covers multiple areas including OLTP online transaction processing with the PC-C and TPC-E sets. Business Intelligence is represented by the warehouse TPC-H benchmark with non-trivial fixed schema and arbitrary scale factor 1GB to 100TB. There is also the database oriented TPC-DS benchmarks featuring nontrivial processing. The TPCx-HS Benchmark [31] is aimed at Hadoop Systems. These benchmarks correspond to OGRE facets AV2, EV10 DV1, 2, 4, PV3, 6.

3.2.3. Enterprise Database Benchmarks: BigBench

BigBench [32, 33] is an industry-led effort to define a comprehensive Big Data benchmark that emerged with a proposal appearing in the first workshop on Big Data benchmarking (WBDB) [34]. It is a paper and pencil specification, but comes with a reference implementation to get started. BigBench models a retailer and benchmarks 30 queries around it covering 5 business categories depicted in the McKinsey report [35]. The retailer data model in BigBench addresses the three V's – volume, variety, and velocity – of Big Data systems. It covers variety by introducing structured, semi-structured, and unstructured data in the model. While the first is an adaptation from the TPC-DS benchmark's data model, the semi-structured data represents the click stream on the site, and unstructured data denotes product reviews submitted by users. Volume and velocity are covered with a scale factor in the specification. BigBench is aimed at modern parallel databases like Hive Impala and Shark and covers Ogre facets AV2, EV4-6,9,10 DV1, 2, 4, PV3, 6.

3.2.4. Enterprise Database Benchmarks: Yahoo Cloud Serving Benchmark

This Yahoo cloud serving benchmark [18] benchmarks basic (CRUD) operations (insert, read, update, delete, scan) store for major NoSQL key-value systems Accumulo, Cassandra, Dynamo, HBase, HyperTable, JDBC, MongoDB, and Redis Voldemort. This exhibits Ogre Facets DV1,4 and PV6.

3.2.5. Enterprise Database Benchmarks: Berkeley Big Data Benchmark

The Berkeley Big Data Benchmark [21] investigates parallel SQL and Hadoop environments: Redshift, Hive, Shark, Impala, Stinger/Tez. It takes workloads and 4 distinct SQL style queries from an earlier work from Brown University and collaborators (called CALDA) that produced a similar Hadoop benchmark [36, 37]. This shows Ogre Facets AV2,12 EV9,10 DV1,2,4, PV3,6.

3.2.6. HiBench and Hadoop Oriented Benchmarks from Database to Analytics

Here we summarize several Hadoop oriented benchmarks with HiBench [17, 38] as the most comprehensive. It has five components including:

1. Micro benchmarks including sort, WordCount, and TeraSort, which is a Hadoop-based sort benchmark [11] from Apache. Further the HDFS DFSIO benchmark [39] is enhanced as EnhancedDFSIO.
2. HiBench includes a Web search benchmark built around Apache Nutch (web crawler) Indexing, and PageRank.
3. It includes some Machine learning with Bayesian Classification (training) and K-means Clustering from Apache Mahout [40]
4. It has OLAP analytical query with Join and Aggregation from Hive performance benchmark [41].
5. HiBench recently added an ETL-Recommendation Pipeline, which updates structured web sales data and the unstructured web logs data, and then recalculates the up-to-date item-item similarity matrix, which is the core of online recommendation. [42]

Other Hadoop benchmarks include one [43] from IBM that includes Terasort and the trace-based SWIM [44, 45] (Statistical Workload Injector for MapReduce), a benchmark representing a real-world big data workload developed by University of California at Berkley in close cooperation with Facebook. Gridmix [46] is another Hadoop trace-based benchmark and Terasort is extended [47] with several related benchmarks testing I/O subsystems: GraySort, MinuteSort, and CloudSort. The work of [48] seems similar to the analytics and HDFS side of HiBench. Indiana University has several papers on benchmarks of Iterative and classic Mapreduce extending the analytics side of HiBench and merging with current Facet analysis [49-55]. The benchmarks in this subsection exhibit facets AV2,3,7,8,12 EV9,10, DV1,2,4 and PV1,3,5,6,7,8,9,12.

3.2.7. Integrated Datasystem Benchmarks: BigDataBench

The integrated BigDataBench suite from China [19] has a growing number of components, with version 3.1 covering search, social networks, e-commerce, multimedia and bioinformatics domains. Kernels and micro-benchmarks include database read, write, scan, sort, grep, wordcount, BFS breadth first search, index, PageRank, Kmeans, connected components, collaborative filtering, naive Bayes and Bioinformatics SAND and BLAST. BigDataBench is hosted on Hbase, MySQL, Nutch, MPI, Spark, Hadoop, Hive, Shark, and Impala. Facets probed are AV1,2,3,4,7,8,12 EV11, DV1,2 and PV1,2,3,5,6,7,8,9,11,12,13.

3.2.8. Integrated Datasystem Benchmarks: CloudSuite

The Cloudsuite [22, 56] benchmark collection from Europe has some distinctive features including use of the Faban (from SPEC) [57] workload generator, a simulator and provision of benchmarks as ready to go virtual machine images. It covers data analytics based on a standard Wikipedia Mahout+Hadoop benchmark with Bayes Classifier plus graph analytics TunkRank from GraphLab [58]. Data Caching has a streaming simulated Twitter test using memcached but not Apache Storm. Data serving is based on Yahoo work in section 3.2.4 while other applications include media streaming, software testing, web search and web serving. Software covered includes Darwin, Cloud9, Nutch, Tomcat, Nginx, Olio and MySQL. Facets include AV1,2,4,7,8,12 EV9,10,12, DV 1,2,4 and PV1,2,3,6,7,9,13.

3.3. Machine Learning, Graph and Other Benchmarks

The processing view has the well-known Graph500 [26] benchmarks (and associated machine ranking), but of course libraries like R [59], Mahout [40] and MLlib [60] also include many candidates for analytics benchmarks. Section 3.1 covered a rich set of analytics and 3.2 largely database benchmarks (with some modest analytics) and here we cover other analytics benchmarks. The benchmarks in section 3.3 exhibit Ogre facets AV2,3,4,6,7,8 EV12, DV2,4 and PV2,3,7,8,9,12,13.

3.3.1. Graph500 Benchmarks

There are [26] currently two kernels and 6 sizes from 17GB to 1.1PB which are used to produce the Graph 500 ranking of supercomputers. The first kernel constructs a tree and the second does a breadth first search (BFS). This covers facets AV2,4,6,7,8, EV4 and PV1,3,13. Note that there are several excellent libraries with a rich set of graph algorithms including Oracle PGX [61], GraphLab [58], Intel GraphBuilder [62], GraphX [63], CINET [64], and Pegasus [65].

3.3.2. Minebench

Minebench [27] is a comprehensive data-mining library with 15 members covering five categories: classification, clustering, association rule mining, structure learning and optimization. OpenMP implementations are given for many kernels. There are also specialized machine learning libraries such as Caffe [66], Torch [67] and Theano [68] for deep learning that can form the basis of benchmarks.

3.3.3. BG and LinkBench Benchmarks

BG [28] emulates read and write actions performed on a social networking datastore, mimicking small transactions on Facebook, and benchmarks them against a given service level agreement. LinkBench [29], developed by Facebook, is intended to serve as a synthetic benchmark to predict the performance of a database system serving Facebook's production data.

4. Illustrating Ogres with Initial Benchmarking

4.1. SPIDAL Codes

This section looks at two SPIDAL clustering codes GML1 and GML2 of Table 2 corresponding to metric and non-metric space scenarios [69]. Both use deterministic annealing DA and are believed to be the best available codes for cases when accurate clusters are needed [70] – non-metric DA pairwise clustering (DA-PWC) [71] and the metric DA vector sponge (DA-VS) [1, 72]. Both were originally written in C# and built on MPI.NET and threads running on Windows HPC clusters. They now have been converted to Java and actually get better performance sequentially and in parallel than the original C# versions. More details are available online [73] and the parallel performance of DA-VS has been presented in detail for C# version [74]. We use DA-VS to motivate Micro-benchmarks but focus on DA-PWC here.

4.2. Benchmarking Environment

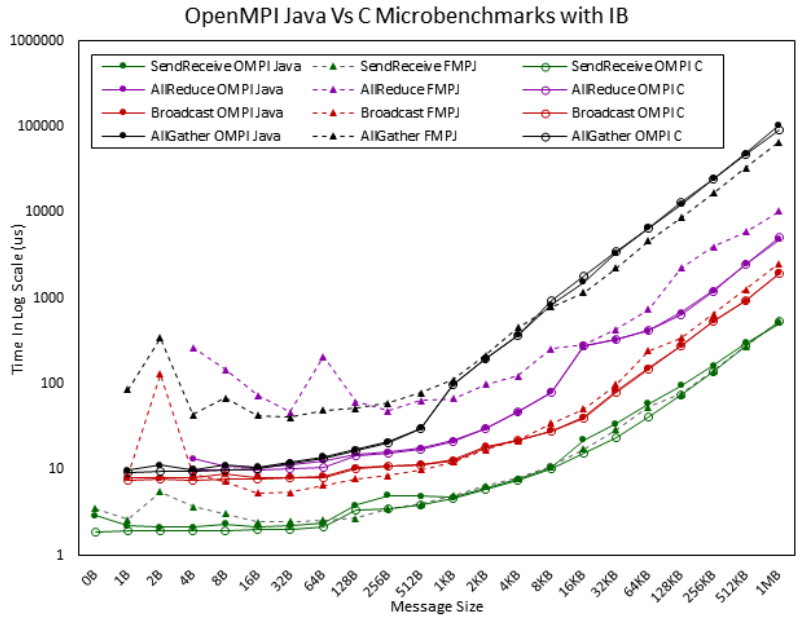

Figure 2: Comparison of OpenMPI 1.8.1 C, OpenMPI 1.8.1 Java and FastMPJ 1.0_6. We evaluate four MPI Collectives: SendReceive, Broadcast, AllReduceand AllGather. Different MPI implementations with message size ranging from 0 bytes (B) up to one megabyte (MB). These are averaged values over patterns 1x1x8, 1x2x8, and 1x4x8 where pattern format is number of concurrent tasks (CT) per process x number of processes per node x number of nodes (i.e. TxPxN).

We used two IU School of Informatics and Computing clusters, Madrid and Tempest, and one FutureGrid [75] cluster – India, as described below.

- **Tempest:** 32 nodes, each has 4 Intel Xeon E7450 CPUs at 2.40GHz with 6 cores, totaling 24 cores per node; 48 GB node memory and 20Gbps Infiniband (IB) network connection. It originally ran Windows Server 2008 R2 HPC Edition – version 6.1 (Build 7601: Service Pack 1). Currently it runs Red Hat Enterprise Linux release version 5.11 (Tikanga)
- **Madrid:** 8 nodes, each has 4 AMD Opteron 8356 at 2.30GHz with 4 cores, totaling 16 cores per node; 16GB node memory and 1Gbps Ethernet network connection. It runs Red Hat Enterprise Linux Server release 6.5
- **India cluster on FutureGrid (FG):** 128 nodes, each has 2 Intel Xeon X5550 CPUs at 2.66GHz with 4 cores, totaling 8 cores per node; 24GB node memory and 20Gbps IB network connection. It runs Red Hat Enterprise Linux Server release 5.10.

Both our clustering codes are written to use a mix of MPI and Thread parallelism and used Microsoft TPL for thread parallelism in a C# .NET 4.0 and MPI.NET 1.0.0 environment. There was no consensus Java OpenMP package we could use to replace TPL for Java codes. We chose to use a novel Java parallel tasks library called Habanero Java (HJ) library from Rice University [76, 77], which requires Java 8.

4.3. Micro-benchmarks

One important issue for data analytics is that many important codes are not in the

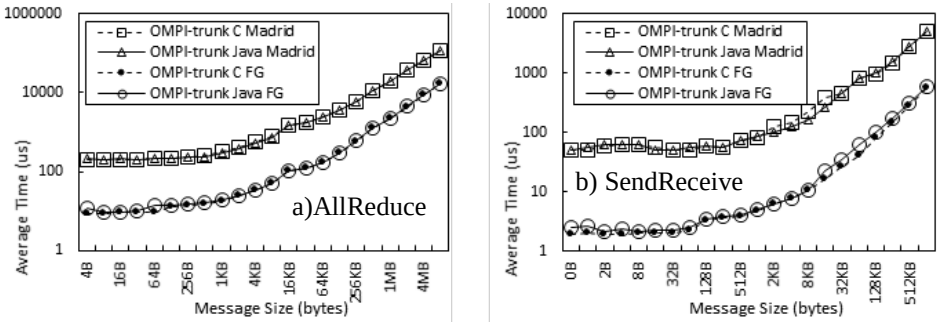

Figure 3: Comparison of MPI performance on machines with Infiniband (FG FutureGrid) and without an Infiniband Network (Madrid) for MPI a) AllReduce and b) SendReceive. These are averaged values over patterns 1x1x8, 1x2x8, and 1x4x8 where pattern format is number of concurrent tasks (CT) per process x number of processes per node x number of nodes (i.e. TxPxN)

C++/Fortran ecosystem familiar from HPC. There are in particular no well-established Java technologies for the core parallel computing technologies MPI and OpenMP. There have been several message passing frameworks for Java [78], but the choice is restricted if you need support for Infiniband (IB) network as discussed in [79]. The situation has clarified recently as OpenMPI now has an excellent Java binding [80], which is an adaptation from the original mpiJava library [81]. This uses wrappers around their C MPI library and we also evaluate FastMPJ 1.0_6, which is a pure Java implementation of mpiJava 1.2 [82] specification and supports IB as does OpenMPI where we use version 1.8.1 unless specified differently.

The SPIDAL codes DA-VS and DA-PWC code rely heavily on a few MPI operations – AllReduce, SendReceive, Broadcast, and AllGather. We studied their Java performance against native implementations with micro-benchmarks adapted from OSU micro-benchmarks suite [83]. Initially C outperformed Java but around November 2013, support to represent data for communications as Java direct buffers (outside the managed heap) was added (around OpenMPI trunk r30301) avoiding earlier Java JNI and object serialization costs resulting in similar performance between Java and C versions of MPI, as reported in figure 2. Note that FastMPJ has good performance for large messages but especially for AllReduce, and is significantly slower than OpenMPI for messages below 1 KB in size.

Figures 3 (a) and (b) show MPI AllReduce and SendReceive performance with and without Infiniband IB. While the decrease in communication times with IB is as expected, the near identical performance of Java with native benchmark in both IB and Ethernet cases is promising for the goal of high performance Java libraries. These figures use OMPI-trunk (r30301), which is older than OMPI 1.8.1 but has similar characteristics because of the fact that it was using direct buffers, which are the main improvement over earlier versions of OpenMPI.

4.4. SPIDAL Clustering Benchmarks

Here we do not discuss FastMPJ as our Java DA-PWC implementation gave frequent runtime errors with this MPI version. We have performed [73] an extensive performance analysis of the two clustering codes comparing C# and Java and looking at

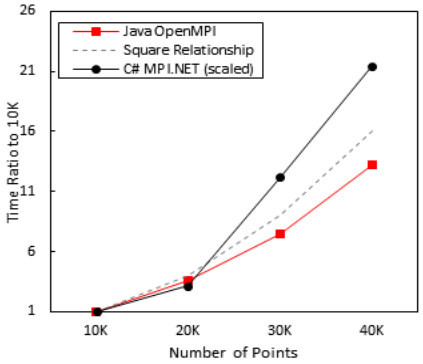

Figure 4: DA-PWC performance as a function of problem size for C# and Java. The Java results come from FutureGrid and C# from Tempest with C# results scaled by 0.7 to reflect measured relative sequential performance of machines. We used 32 nodes each running with 8 way parallelism (MPI internally and between nodes) totaling 256 way parallelism.

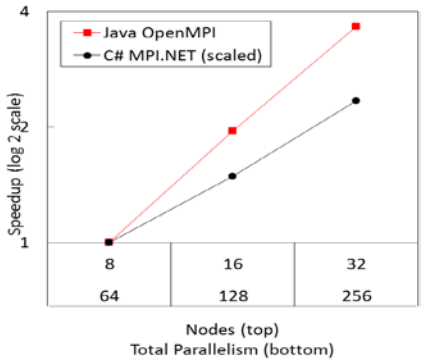

Figure 5: Speedup of 40K point DA-PWC on 8, 16 and 32 nodes for case where 8 MPI processes run on each node. . The Java results come from FutureGrid and C# from Tempest with C# results scaled by 0.7. Results are scaled to performance of 8 node run.

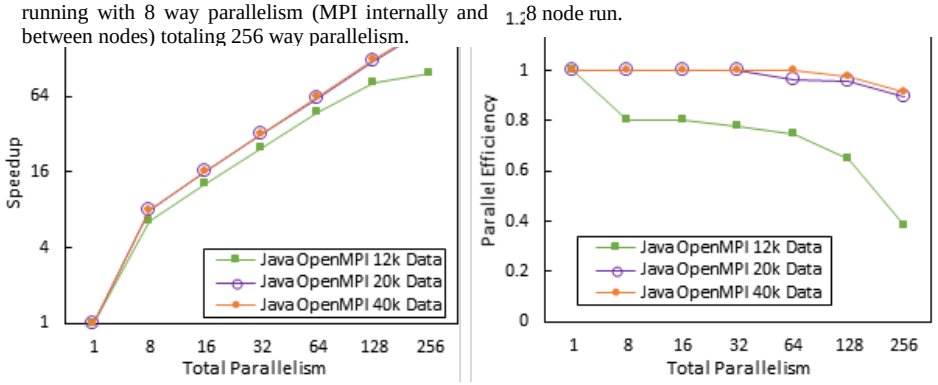

Figure 6: Strong Scaling expressed as speedup and efficiency on FutureGrid for 3 datasets (12K, 20K, 40K) as a function of parallelism from 1 to 256. Apart from sequential case, all runs ran 8-way parallel on each node; results are averaged over that node parallelism being 8 MPI processes, 2 threads, 4 MPI processes or 4 threads, 2 MPI processes

different parallelism choices in the nodes; MPI or threading. We give some illustrative results here. DA-PWC should scale in execution time like problem size squared and Figure 4 shows Java results consistent with this; the 40K problem size runs faster than 16 times 10K execution time due to increased communication on a smaller problem. The C# results are not consistent with the expected model and illustrative of other anomalies we found with C#. We did not explore more as Microsoft has abandoned this platform. Figure 5 examines this for fixed problem size (strong scaling) increasing parallelism from 64 to 256.

Figure 6 summarizes speedups and parallel efficiencies for all datasets across parallelisms from 1 to 256. The intention of this is to illustrate the behavior with increasing parallelism for different data sizes. It shows that DA-PWC scales well with

the increase in parallelism up to the limit of 256 for 20K and 40K data sets. The 12K data set shows reasonable scaling within 8 and 64 way parallelisms, but not outside this range. This is the usual issue that small problems increase their communication fraction as you increase parallelism with strong scaling.

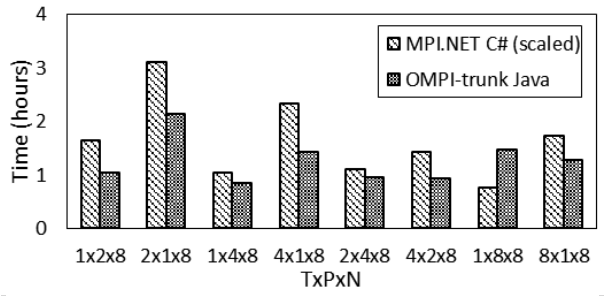

Figure 7: Comparison of C# and Java with MPI and Threads for DA-VS SPIDAL clustering of 240K points and about 25000 clusters studied in [1]. The Java results come from FutureGrid OpenMPI trunk r30301 and C# from Tempest with C# results scaled by 0.7 to reflect measured relative sequential performance of machines. The runs are labelled TxPxN where T threads and P MPI processes are run in each of N nodes.

Figures 7 and 8 compare the use of threads and MPI process with runs labelled TxPxN where T threads and P MPI processes are run in each of N nodes. In figure 7, we compare PxQx8 with QxPx8 for (P,Q) choices (1,2) (1,4) (2,4) (1,8) with best performance occurring at 1x8x8 for C# and 1x4x8 for Java. Figure 8 looks at DA-PWC with parallelisms from 1 to 32 realized in different choices between threads and MPI. There are a set of 6 speedup groups for the same parallelism – 1, 2, 4, 8, 16, and 32. These lead to plateaus in plot corresponding to MPI and threads giving similar speedups, except for the 8x1xN cases, which shows lower performance. This effect occurs as the machine (India) has 2 physical CPUs each with 4 cores, so running 1 process with more than 4 concurrent tasks appears to introduce thread switches between CPUs, which is expensive.

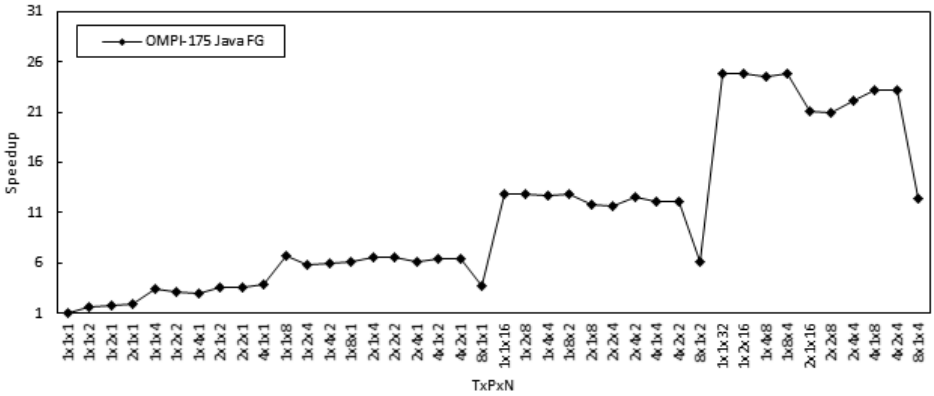

Figure 8: Speed up of DA-PWC for a variety of threading/MPI parallelism choices from sequential through 32 way parallelism on the 12K dataset. The runs are labelled TxPxN where T threads and P MPI processes are run in each of N nodes. These runs are on FutureGrid using OpenMPI 1.7.5 which is similar in performance to 1.8.1

The best approach in this case is to restrict number of threads to be <= number of threads per CPU and use MPI across CPUs. We note that DA-PWC is a simpler algorithm than DA-VS and gets more reliable answers. We intend to run DA-VS on much larger datasets.

This section has shown that data analytics performance can be analyzed quantitatively using techniques familiar from parallel computing. One needs to use the

execution facets EV2-5 to specify the parameters of the testing environment. The trade-off between MPI and thread parallelism needs attention, as does use of GPUs not covered here.

5. Ogre-Driven Benchmarking

First, we note some qualitatively different types of benchmarks. There are at the simplest level "micro-benchmarks" PV1 which capture core machine performance. Then we have "atomic" kernels – simple non-trivial algorithms or problems that cannot usefully be broken up. Then we see a class we call "mini-apps" that are the most complex and can be constructed in two different ways: top-down and bottom-up. In the top-down case, we start with a real complex application and simplify it to capture some "key" capabilities but we do not necessarily make it "atomic". In the bottom-up approach, we take multiple "atomic kernels" and link them together such as in benchmarking Mahout's clustering algorithm reading data from Hbase. This comment is relevant as most of the facets not covered in Table A would be addressed by composite Ogre instances, which for example would cover Fusion and Dataflow in Problem Architecture view AV9, 10. One striking area not covered in Section 3 is streaming with facets AV-5 DV-8 and PV-10. Further veracity (EV-7), the object store – file comparison (DV3), metadata (DV7) and HPC simulations DV-9 all certainly need to be addressed.

Our suggested process is to choose benchmarks that cover a broad range of facets. As discussed above and seen in Tables 1 and A, the currently identified benchmarks do not cover some of the facets. This can be explained by the fact that facets come from an analysis of full applications and Table 1 has by design only analytics kernels. We expect that an addition of a few extra "atomic" Ogre instances and a set of mini-apps or composite instances will lead to a rather good facet coverage. There is quite a lot of redundancy in current benchmarks and here again the strategy of covering but not over covering all facets provides systematics principles.

One must also address the many well-studied general points of benchmarking, such as agreeing on datasets with various sizes (Volume facet in Execution view), requiring correct answers for each implementation, and the choice between pencil and paper and source code specification of a benchmark.

6. Appendix Table A: Detailed Facet Listing with Benchmark Coverage

Facet and View		Comments	SP	DB	NI
Facets in Problem Architecture View (AV)					
1	Pleasingly Parallel	Clear qualitative property overlapping Local Analytics	M	S	H
2	Classic MapReduce	Clear qualitative property of non-iterative algorithms	M	H	H
3	Map-Collective	Clear qualitative property of much machine learning	H	S	H
4	Map Point-to-Point (graphs)	Clear qualitative property of graphs and simulation	H	S	M
5	Map Streaming	Property of growing importance. Not well benchmarked	N	N	H
6	Shared memory (as opposed to distributed parallel algorithm)	Corresponds to problem where shared memory implementations important. Tend to be dynamic asynchronous	S	N	S

7	Single Program Multiple Data SPMD	Clear qualitative property famous in parallel computing	H	M	H
8	Bulk Synchronous Processing BSP	Needs to be defined but reasonable qualitative property	H	M	H
9	Fusion	Only present for composite Ogres	N	N	H
10	Dataflow	Only present for composite Ogres	N	N	H
11	Agents	Clear but uncommon qualitative property	N	N	S
12	Orchestration (workflow)	Only present for composite Ogres	N	H	H
	Facets in Execution View (EV)				
1	Performance Metrics	Result of Benchmark	-	-	-
2	Flops per Byte (Memory or I/O). Flops per watt (power).	I/O Not needed for "pure in memory" benchmark. Value needs detailed quantitative study. Could depend on implementation	-	-	-
3	Execution Environment (LN = Libraries needed, C= Cloud, HPC = HPC, T=Threads, MP= Message Passing)	Depends on how benchmark set up. Could include details of machine used for benchmarking here	-	-	-
4	Volume	Depends on data size. Benchmark measure	-	M	-
5	Velocity	Associated with streaming facet but value depends on particular problem	N	S	H
6	Variety	Most useful for composite Ogres	N	S	H
7	Veracity	Most problems would not discuss but potentially important	N	N	M
8	Communication Structure (D=Distributed, I=Interconnect, S=Synchronization)	Qualitative property – related to BSP and Shared memory	U	U	U
9	D=Dynamic or S=Static	Clear qualitative properties. Importance familiar from parallel computing	H	H	H
10	R=Regular or I=Irregular		H	H	H
11	Iterative?	Clear qualitative property. Highlighted by Iterative MapReduce and always present in classic parallel computing	H	S	H
12	Data Abstraction(K= key-value, BW= bag of words, BI = bag of items, P= pixel/spatial, V= vectors/matrices, S= sequence, G= graph)	Clear quantitative property although important data abstractions not agreed upon. All should be supported by Programming model and run time	H	M	H
13	M= Metric Space or N= not?	Clear qualitative property discussed in [69]	H	N	H
14	NN= O(N2) or N= O(N)?	Clear qualitative property highlighted in [3]	H	N	H
	Facets in Data Source & Style View (DV)				
1	SQL/NoSQL/NewSQL?	Clear qualitative property. Can add NoSQL sub-categories such as key-value, graph, document …	N	H	H
2	Enterprise data model (warehouses)	Clear qualitative property of data model highlighted in database community / industry benchmarks	N	H	M
3	Files/Objects?	Clear qualitative property of data model where files important in Science; objects in industry	N	S	H
4	HDFS/Lustre/GPFS?	Clear qualitative property where HDFS important in Apache stack but not much used in science	N	H	H
5	Archive/Batched/Streaming	Clear qualitative property but not for kernels as it describes how data is collected	N	N	H
6	Shared/Dedicated/Transient/Permanent	Clear qualitative property of data whose importance is not well studied	N	N	H
7	Metadata/Provenance	Clear qualitative property but not for kernels as important aspect of data collection process	N	N	H
8	Internet of Things	Dominant source of commodity data in future	N	N	H
9	HPC Simulations	Important in science research especially at exascale	N	N	H
10	Geographic Information Systems	Clear property but not for kernels	S	N	H

Facets in Processing View (PV)					
1	Micro-benchmarks	Important subset of small kernels	N	H	N
2	Local Analytics or Informatics	Well defined but overlaps Pleasingly Parallel	H	H	H
3	Global Analytics or Informatics	Clear qualitative property that includes parallel Mahout (E.g. Kmeans) and Hive (database)	H	H	H
4	Base Statistics	Describes simple statistical averages needing simple MapReduce. MRStat in [7]	N	N	M
5	Recommender Engine	Clear type of machine learning of especial importance commercially	N	M	H
6	Search/Query/Index	Clear important class of algorithms in industry	S	H	H
7	Classification	Clear important class of algorithms	S	M	H
8	Learning	Includes deep learning as category	S	S	H
9	Optimization Methodology (ML= Machine Learning, NO = Nonlinear Optimization, LS = Least Squares, EM = expectation maximization, LQP = Linear/Quadratic Programming, CO = Combinatorial Optimization)	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)	H	M	H
10	Streaming	Clear important class of algorithms associated with Internet of Things. Can be called DDDAS Dynamic Data-Driven Application Systems	N	N	H
11	Alignment	Clear important class of algorithms as in BLAST	N	S	M
12	Linear Algebra Kernels	Important property of some analytics	H	S	H
13	Graph Algorithms	Clear important class of algorithms – often hard	H	M	M
14	Visualization	Clearly important aspect of data analysis but different in character to most other facets	S	N	H

Table A: The four views and their constituent facets: H: High Use; M: Medium use; S: Small use; N: essentially no use; - inapplicable; U: Unknown. SP is SPIDAL project of section 3.1 [13]. DB is Database analysis of section 3.2 [15]. NI is NIST Public Working Group Use Case Working Group analysis [7]

References

[1] Geoffrey Fox, D. R. Mani, and Saumyadipta Pyne, *Parallel Deterministic Annealing Clustering and its Application to LC-MS Data Analysis*, in *IEEE International Conference on Big Data*. October 6-9, 2013. Santa Clara, CA, USA, . DOI: http://dx.doi.org/10.1109/BigData.2013.6691636.

[2] Asanovic, K., R. Bodik, B.C. Catanzaro, J.J. Gebis, P. Husbands, K. Keutzer, D.A. Patterson, W.L. Plishker, J. Shalf, S.W. Williams, and K.A. Yelick. *The Landscape of Parallel Computing Research: A View from Berkeley*. 2006 December 18 [accessed 2009 December]; Available from: http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.html.

[3] Committee on the Analysis of Massive Data; Committee on Applied and Theoretical Statistics; Board on Mathematical Sciences and Their Applications; Division on Engineering and Physical Sciences; National Research Council, *Frontiers in Massive Data Analysis*. 2013: National Academies Press. http://www.nap.edu/catalog.php?record_id=18374

[4] Shantenu Jha, Judy Qiu, Andre Luckow, Pradeep Mantha, and Geoffrey C. Fox, *A Tale of Two Data-Intensive Approaches: Applications, Architectures and Infrastructure*, in *3rd International IEEE Congress on Big Data Application and Experience Track*. June 27- July 2, 2014. Anchorage, Alaska. http://arxiv.org/abs/1403.1528.

[5] Geoffrey C.Fox, Shantenu Jha, Judy Qiu, and Andre Luckow, *Towards an Understanding of Facets and Exemplars of Big Data Applications*, in *20 Years of Beowulf: Workshop to Honor Thomas Sterling's 65th Birthday* October 14, 2014. Annapolis http://grids.ucs.indiana.edu/ptliupages/publications/OgrePaperv9.pdf.

[6] Judy Qiu, Shantenu Jha, Andre Luckow, and Geoffrey C.Fox, *Towards HPC-ABDS: An Initial High-Performance Big Data Stack*, in *Building Robust Big Data Ecosystem ISO/IEC JTC 1 Study Group on Big Data*. March 18-21, 2014. San Diego Supercomputer Center, San Diego. http://grids.ucs.indiana.edu/ptliupages/publications/nist-hpc-abds.pdf.

[7] NIST, *NIST Big Data Public Working Group (NBD-PWG) Use Cases and Requirements*. 2013. http://bigdatawg.nist.gov/usecases.php

[8] Geoffrey Fox and Wo Chang, *Big Data Use Cases and Requirements*, in *1st Big Data Interoperability Framework Workshop: Building Robust Big Data Ecosystem ISO/IEC JTC 1 Study Group on Big Data* March 18 - 21, 2014. San Diego Supercomputer Center, San Diego. http://grids.ucs.indiana.edu/ptliupages/publications/NISTUseCase.pdf.

[9] NASA Advanced Supercomputing Division. *NAS Parallel Benchmarks*. 1991 [accessed 2014 March 28]; Available from: https://www.nas.nasa.gov/publications/npb.html.

[10] A. Petitet, R. C. Whaley, J. Dongarra, and A. Cleary. *HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers*. 2008 September 10 [accessed 2014 July 19,]; Available from: http://www.netlib.org/benchmark/hpl/.

[11] Apache Hadoop. *TeraSort map/reduce sort*. [accessed 2015 January 12]; Available from: http://hadoop.apache.org/docs/current/api/org/apache/hadoop/examples/terasort/package-summary.html.

[12] *SPEC's Benchmarks*. [accessed 2015 January 12]; Available from: https://www.spec.org/benchmarks.html.

[13] Indiana, Rutgers, Virginia Tech, Kansas, Stony Brook, Arizona State, and Utah. *CIF21 DIBBs: Middleware and High Performance Analytics Libraries for Scalable Data Science Summary*. 2014 December 18 [accessed 2015 January 12]; Available from: http://grids.ucs.indiana.edu/ptliupages/presentations/Dibbs%20-%20Overall%20-%20Dec18-2014.pptx

[14] *HPC-ABDS Kaleidoscope of over 270 Apache Big Data Stack and HPC Tecnologies*. [accessed 2014 April 8]; Available from: http://hpc-abds.org/kaleidoscope/.

[15] Chaitan Baru and Tilmann Rabl. *Tutorial 4 " Big Data Benchmarking" at 2014 IEEE International Conference on Big Data*. 2014 [accessed 2015 January 2]; Available from: http://cci.drexel.edu/bigdata/bigdata2014/tutorial.htm.

[16] Transaction Processing Performance Council. *TPC Benchmarks*. [accessed 2015 January 12]; Available from: http://www.tpc.org/information/benchmarks.asp.

[17] Lan Yi, *Experience with HiBench: From Micro-Benchmarks toward End-to-End Pipelines*, in *Third Workshop on Big Data Benchmarking,WBDB*. July 16-17, 2013. Xi'an China. http://clds.ucsd.edu/sites/clds.ucsd.edu/files/HiBench-wbdb-2013-07-16-xian-yilan-intel.ppt.

[18] Brian Cooper. *Yahoo Cloud Serving Benchmark with Key-value store (Accumulo, Cassandra, Hbase, MongoDB etc.) benchmarks*. [accessed 2015 January 12]; Available from: https://github.com/brianfrankcooper/YCSB/

[19] Jianfeng Zhan. *BigDataBench: A Big Data Benchmark Suite from ICT, Chinese Academy of Sciences*. [accessed 2015 January 12]; Available from: http://prof.ict.ac.cn/BigDataBench/.

[20] Tilmann Rabl. *Big Data Analytics Benchmark (BigBench)*. [accessed 2015 January 12]; Available from: http://www.msrg.org/projects/project_view.php?id=12.

[21] AMPLab. *Berkeley Big Data Benchmark: Redshift, Hive, Shark, Impala, Stinger/Tez benchmarks*. [accessed 2015 January 12]; Available from: https://amplab.cs.berkeley.edu/benchmark/.

[22] *CloudSuite benchmark suite for emerging scale-out applications*. [accessed 2015 February 13]; Available from: http://parsa.epfl.ch/cloudsuite/cloudsuite.html.

[23] SPEC Standard Performance Evaluation Corporation. *SPEC's Benchmarks*. [accessed 2015 February 13]; Available from: https://www.spec.org/benchmarks.html.

[24] H. Howie Huang, Shan Li, Alex Szalay, and Andreas Terzis, *Performance modeling and analysis of flash-based storage devices*, in *Proceedings of the 2011 IEEE 27th Symposium on Mass Storage Systems and Technologies*. 2011, IEEE Computer Society. pages. 1-11. DOI: 10.1109/msst.2011.5937213.

[25] Nusrat Sharmin Islam, Xiaoyi Lu, Md. Wasi-ur-Rahman, Jithin Jose, and Dhabaleswar K. Panda, *A Micro-benchmark Suite for Evaluating HDFS Operations on Modern Clusters*, Chapter 12 in *Specifying Big Data Benchmarks*, T. Rabl, M. Poess, C. Baru, and H.-A. Jacobsen, Editors. 2014, Springer Berlin Heidelberg. p. 129-147. http://dx.doi.org/10.1007/978-3-642-53974-9_12.

[26] *Graph 500 Benchmarks*. [accessed 2015 January 12]; Available from: http://www.graph500.org/specifications.

[27] Ramanathan Narayanan, Berkin Ozisikyilmaz, Joseph Zambreno, Gokhan Memik, Alok Choudhary, and Jayaprakash Pisharath. *MineBench: A Benchmark Suite for Data Mining Workloads*. in *2006 IEEE International Symposium on Workload Characterization*. 25-27 Oct. 2006 2006.

[28] Barahmand, S. and S. Ghandeharizadeh, *BG: A Benchmark to Evaluate Interactive Social Networking Actions*, in *CIDR*. 2013, www.cidrdb.org. http://dblp.uni-trier.de/db/conf/cidr/cidr2013.html#BarahmandG13.

[29] Armstrong, T.G., V. Ponnekanti, D. Borthakur, and M. Callaghan, *LinkBench: a database benchmark based on the Facebook social graph*, in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. 2013, ACM. New York, New York, USA. pages. 1185-1196. DOI: 10.1145/2463676.2465296.

[30] SPEC Standard Performance Evaluation Corporation. *SPEC Big Data Working Group*. [accessed 2015 February 13]; Available from: http://research.spec.org/working-groups/big-data-working-group.html.

[31] Transaction Processing Performance Council. *TPC Express Benchmark HS (TPCx-HS) for Hadoop* [accessed 2015 February 13]; Available from: http://www.tpc.org/tpcx-hs/.
[32] Ghazal, A., T. Rabl, M. Hu, F. Raab, M. Poess, A. Crolotte, and H.-A. Jacobsen, *BigBench: towards an industry standard benchmark for big data analytics*, in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. 2013, ACM. New York, New York, USA. pages. 1197-1208. DOI: 10.1145/2463676.2463712.
[33] Gowda, B.D. and N. Ravi. *BigBench: Toward An Industry-Standard Benchmark for Big Data Analytics*. 2014 November 25, 2014 1/4/15]; Available from: http://blog.cloudera.com/blog/2014/11/bigbench-toward-an-industry-standard-benchmark-for-big-data-analytics/.
[34] Tilmann Rabl, Meikel Poess, Chaitanya Baru, and H.-A. Jacobsen. *Specifying Big Data Benchmarks: First Workshop, WBDB 2012, San Jose, CA, USA, May 8-9, 2012 and Second Workshop, WBDB 2012, Pune, India, December 17-18, 2012, Revised Selected Papers* 2014: Springer Berlin Heidelberg.
[35] Manyika, J., M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A.H. Byers, *Big data: The next frontier for innovation, competition, and productivity*. May 2011, 2011. http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation.
[36] Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden, and Michael Stonebraker, *A comparison of approaches to large-scale data analysis*, in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. 2009, ACM. Providence, Rhode Island, USA. pages. 165-178. DOI: 10.1145/1559845.1559865.
[37] Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden, and Michael Stonebraker. *A Comparison of Approaches to Large-Scale Data Analysis*. 2011 August 24 [accessed 2015 February 13]; Available from: http://database.cs.brown.edu/projects/mapreduce-vs-dbms/.
[38] Shengsheng Huang, Jie Huang, Jinquan Dai, Tao Xie, and Bo Huang, *The HiBench Benchmark Suite: Characterization of the MapReduce-Based Data Analysis*, Chapter 9 in *New Frontiers in Information and Software as Services*, D. Agrawal, K.S. Candan, and W.-S. Li, Editors. 2011, Springer Berlin Heidelberg. p. 209-228. http://dx.doi.org/10.1007/978-3-642-19294-4_9.
[39] E. Paulson. *HDFS Benchmarks*. [accessed 2015 January 11]; Available from: http://epaulson.github.io/HadoopInternals/benchmarks.html#dfsio.
[40] *Apache Mahout Scalable machine learning and data mining* [accessed 2012 August 22]; Available from: http://mahout.apache.org/.
[41] Apache. *Hive performance benchmarks*. [accessed 2015 February 14]; Available from: https://issues.apache.org/jira/browse/HIVE-396.
[42] *Intel Hadoop Benchmark for ETL-Recommendation Pipeline* [accessed 2015 February 13]; Available from: https://github.com/intel-hadoop/HiBench/tree/etl-recomm.
[43] IBM. *Apache Hadoop vs. IBM Platform Symphony and Infosphere* [accessed 2015 February 13]; Available from: http://www-03.ibm.com/systems/platformcomputing/products/symphony/highperfhadoop.html.
[44] Berkeley-Facebook. *Statistical Workload Injector for MapReduce (SWIM).* [accessed 2015 February 13]; Available from: https://github.com/SWIMProjectUCB/SWIM/wiki.
[45] Chen Yanpei, A. Ganapathi, R. Griffith, and R. Katz. *The Case for Evaluating MapReduce Performance Using Workload Suites*. in *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2011 IEEE 19th International Symposium on*. 25-27 July 2011.
[46] Apache Hadoop. *GridMix benchmark for Hadoop clusters*. [accessed 2015 February 13]; Available from: http://hadoop.apache.org/docs/r1.2.1/gridmix.html
[47] Chris Nyberg, Mehul Shah, Naga Govindaraju, and Jim Gray. *Sort Benchmark Home Page*. [accessed 2015 January 3]; Available from: http://sortbenchmark.org/.
[48] Z. Fadika, M. Govindaraju, R. Canon, and L. Ramakrishnan. *Evaluating Hadoop for Data-Intensive Scientific Operations*. in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*. 24-29 June 2012 2012.
[49] Bingjing Zhang, Yang Ruan, and Judy Qiu, *Harp: Collective Communication on Hadoop*, in *IEEE International Conference on Cloud Engineering (IC2E) conference* October 10, 2014. http://grids.ucs.indiana.edu/ptliupages/publications/HarpQiuZhang.pdf.
[50] Thilina Gunarathne, Bingjing Zhang, Tak-Lon Wu, and Judy Qiu, *Scalable Parallel Computing on Clouds Using Twister4Azure Iterative MapReduce* Future Generation Computer Systems 2013. 29(4): p. 1035-1048. http://grids.ucs.indiana.edu/ptliupages/publications/Scalable_Parallel_Computing_on_Clouds_Using_Twister4Azure_Iterative_MapReduce_cr_submit.pdf
[51] Judy Qiu, Jaliya Ekanayake, Thilina Gunarathne, Jong Youl Choi, Seung-Hee Bae, Hui Li, Bingjing Zhang, Tak-Lon Wu, Yang Ryan, Saliya Ekanayake, Adam Hughes, and Geoffrey Fox, *Hybrid cloud and cluster computing paradigms for life science applications.* BMC Bioinformatics, 2010. Proceedings of

BOSC 2010. http://grids.ucs.indiana.edu/ptliupages/publications/HybridCloudandClusterComputingParadigmsforLifeScienceApplications_Pub.pdf

[52] Bingjing Zhang, Yang Ruan, Tak-Lon Wu, Judy Qiu, Adam Hughes, and Geoffrey Fox, *Applying Twister to Scientific Applications*, in *CloudCom 2010*. November 30-December 3, 2010. IUPUI Conference Center Indianapolis. http://grids.ucs.indiana.edu/ptliupages/publications/PID1510523.pdf.

[53] Geoffrey Fox, Seung-Hee Bae, Jaliya Ekanayake, Xiaohong Qiu, and H. Yuan, *Parallel Data Mining from Multicore to Cloudy Grids*. book chapter of High Speed and Large Scale Scientific Computing. 2009: IOS Press, Amsterdam. http://grids.ucs.indiana.edu/ptliupages/publications/CetraroWriteupJune11-09.pdf. ISBN:978-1-60750-073-5

[54] J.Ekanayake, H.Li, B.Zhang, T.Gunarathne, S.Bae, J.Qiu, and G.Fox, *Twister: A Runtime for iterative MapReduce*, in *Proceedings of the First International Workshop on MapReduce and its Applications of ACM HPDC 2010 conference June 20-25, 2010*. 2010, ACM. Chicago, Illinois. http://grids.ucs.indiana.edu/ptliupages/publications/hpdc-camera-ready-submission.pdf.

[55] Bingjing Zhang, Yang Ruan, and Judy Qiu, in *IEEE International Conference on Cloud Engineering (IC2E)*. March 9-12, 2015. Tempe AZ. http://grids.ucs.indiana.edu/ptliupages/publications/HarpQiuZhang.pdf.

[56] *ISCA 2013 Tutorial: CloudSuite 2.0 on Flexus*. [accessed 2015 February 2015]; Available from: http://parsa.epfl.ch/cloudsuite/isca13-tutorial.html.

[57] *Faban open source performance workload creation and execution framework.* [accessed 2015 February 14]; Available from: http://faban.org/.

[58] Yucheng Low, Danny Bickson , Joseph Gonzalez , Carlos Guestrin , Aapo Kyrola , and Joseph M. Hellerstein, *GraphLab: A New Framework For Parallel Machine Learning*, in *The 26th Conference on Uncertainty in Artificial Intelligence (UAI 2010)*. July 8-11, 2010. Catalina Island, California,.

[59] *R open source statistical library*. [accessed 2012 December 8]; Available from: http://www.r-project.org/.

[60] *Machine Learning Library (MLlib)*. [accessed 2014 April 1]; Available from: http://spark.apache.org/docs/0.9.0/mllib-guide.html.

[61] Oracle. *Parallel Graph Analytics (PGX)*. [accessed 2015 February 14]; Available from: http://www.oracle.com/technetwork/oracle-labs/parallel-graph-analytics/overview/index.html.

[62] Intel. *Graphbuilder: Scalable Graph Construction For Big Data Open Source Code* [accessed 2015 February 14]; Available from: https://software.intel.com/en-us/articles/graphbuilder-scalable-graph-construction-for-big-data-open-source-code-release.

[63] Reynold S. Xin, Joseph E. Gonzalez, Michael J. Franklin, and Ion Stoica, *GraphX: a resilient distributed graph system on Spark*, in *GRADES*P.A. Boncz and T. Neumann, Editors., 2013, CWI/ACM. pages. 2. http://dblp.uni-trier.de/db/conf/sigmod/grades2013.html#XinGFS13.

[64] Sherif Elmeligy Abdelhamid, Richard Alo, S. M. Arifuzzaman, Pete Beckman, Md Hasanuzzaman Bhuiyan, Keith Bisset, Edward A. Fox, Geoffrey C. Fox, Kevin Hall, S.M.Shamimul Hasan, Anurodh Joshi, Maleq Khan, Chris J. Kuhlman, Spencer Lee, Jonathan P. Leidig, Hemanth Makkapati, Madhav V. Marathe, Henning S. Mortveit, Judy Qiu, S.S. Ravi, Zalia Shams, Ongard Sirisaengtaksin, Rajesh Subbiah, Samarth Swarup, Nick Trebon, Anil Vullikanti, and Zhao Zhao, *CINET: A CyberInfrastructure for Network Science (NDSSL Technical Report 12-041, Virginia Bioinformatics Institute, Virginia Tech)*, in *8th IEEE International Conference on eScience (eScience 2012)*. October 8-12, 2012, IEEE. Chicago, ILL. http://grids.ucs.indiana.edu/ptliupages/publications/eSciencesArticle.pdf.

[65] U Kang, Charalampos Tsourakakis, and Christos Faloutsos, *PEGASUS: A Peta-Scale Graph Mining System - Implementation and Observations*, in *ICDM*. December, 2009. Miami, Florida.

[66] Berkeley Vision and Learning Center (BVLC) *Caffe deep learning framework*. [accessed 2015 February 14]; Available from: http://caffe.berkeleyvision.org/.

[67] *Torch open source deep learning library for the Lua programming language*. [accessed 2015 February 14]; Available from: http://en.wikipedia.org/wiki/Torch_%28machine_learning%29.

[68] *Python framework for Deep Learning*. [accessed 2015 February 14]; Available from: http://deeplearning.net/software/theano/.

[69] Geoffrey Fox, *Robust Scalable Visualized Clustering in Vector and non Vector Semimetric Spaces.* Parallel Processing Letters, June, 2013. 23(2). DOI:http://www.worldscientific.com/doi/abs/10.1142/S0129626413400069. http://grids.ucs.indiana.edu/ptliupages/publications/Clusteringv1.pdf

[70] Rose, K., E. Gurewwitz, and G. Fox, *A deterministic annealing approach to clustering*. Pattern Recogn. Lett., 1990. 11(9): p. 589-594. DOI:10.1016/0167-8655(90)90010-y

[71] Fox, G.C. and S. Ekanayake. *Deterministic Annealing Pairwise Clustering*. Available from: https://github.com/DSC-SPIDAL/dapwc.

[72] Fox, G.C. and S. Ekanayake. *Deterministic Annealing Vector Sponge*. Available from: https://github.com/DSC-SPIDAL/davs.
[73] Saliya Ekanayake *Indiana University PhD Proposal: Towards a Systematic Approach to Big Data Benchmarking*. 2015 February 2 [accessed 2015 February 11]; Available from: http://grids.ucs.indiana.edu/ptliupages/publications/proposal_final_v2.pdf.
[74] Geoffrey Fox , D. R. Mani, and Saumyadipta Pyne. *Detailed Results of Evaluation of DAVS(c) Deterministic Annealing Clustering and its Application to LC-MS Data Analysis* 2013 June 22 [accessed 2013 June 22]; Available from: http://grids.ucs.indiana.edu/ptliupages/publications/DAVS2.pdf.
[75] von Laszewski, G., G.C. Fox, W. Fugang, A.J. Younge, A. Kulshrestha, G.G. Pike, W. Smith, Vo, x, J. ckler, R.J. Figueiredo, J. Fortes, and K. Keahey. *Design of the FutureGrid experiment management framework*. in *Gateway Computing Environments Workshop (GCE), 2010*. 14-14 Nov. 2010 2010.
[76] Vincent Cavé, Jisheng Zhao, Jun Shirako, and Vivek Sarkar, *Habanero-Java: the new adventures of old X10*, in *Proceedings of the 9th International Conference on Principles and Practice of Programming in Java*. 2011, ACM. Kongens Lyngby, Denmark. pages. 51-61. DOI: 10.1145/2093157.2093165.
[77] Vivek Sarkar and Shams Mahmood Imam. *HJ Library*. Available from: https://wiki.rice.edu/confluence/display/PARPROG/HJ+Library.
[78] Guillermo L. Taboada, Sabela Ramos, Roberto R. Expósito, Juan Touriño, and Ramón Doallo, *Java in the High Performance Computing arena: Research, practice and experience.* Sci. Comput. Program., 2013. 78(5): p. 425-444. DOI:10.1016/j.scico.2011.06.002
[79] Roberto R. Expósito, Sabela Ramos, Guillermo L. Taboada, Juan Touriño, and Ramón Doallo, *FastMPJ: a scalable and efficient Java message-passing library.* Cluster Computing, 2014/02/06, 2014: p. 1-20. DOI:10.1007/s10586-014-0345-4. http://dx.doi.org/10.1007/s10586-014-0345-4
[80] Oscar Vega-Gisbert, J.E.R., Jeffrey M. Squyres, *Design and implementation of Java bindings in Open MPI*. 2014. users.dsic.upv.es/~jroman/preprints/ompi-java.pdf.
[81] Baker, M., B. Carpenter, G. Fox, S. Hoon Ko, and S. Lim, *mpiJava: An object-oriented java interface to MPI*, Chapter 82 in *Parallel and Distributed Processing*, J. Rolim, F. Mueller, A. Zomaya, F. Ercal, S. Olariu, B. Ravindran, J. Gustafsson, H. Takada, R. Olsson, L. Kale, P. Beckman, M. Haines, H. ElGindy, D. Caromel, S. Chaumette, G. Fox, Y. Pan, K. Li, T. Yang, G. Chiola, G. Conte, L.V. Mancini, D. Méry, B. Sanders, D. Bhatt, and V. Prasanna, Editors. 1999, Springer Berlin Heidelberg. p. 748-762. http://dx.doi.org/10.1007/BFb0097964.
[82] Bryan Carpenter, G.F., Sung-Hoon Ko and Sang Lim, *mpiJava 1.2: API Specification*. 1999. https://www.open-mpi.org/papers/mpi-java-spec/mpiJava-spec.pdf.
[83] The Ohio State University's Network-Based Computing Laboratory (NBCL). *OSU Micro-Benchmarks*. Available from: http://mvapich.cse.ohio-state.edu/benchmarks/.