# A brief history and trends in Benchmarking BigData Systems

Frans Ojala: frans.ojala@cs.helsinki.fi, Helsinki University, department of computer science

***The proliferation of the internet to consumers in the last two decades has brought an information explosion. Still some 20 years ago scientists and engineers were only dreaming of data in the peta- and exabyte scales. Today, data in these scales being commonplace, we are in need of larger and more powerful systems than ever to compute on that data. Many such systems have been devised and are in use - Hadoop MapReduce, Spark, Teradata and Cassandra to name a few. But what is the real-world power of new BigData systems? Can we differentiate between systems of various vendors and get the best system for our capital? To be able to answer such questions the systems need to be tested, benchmarked, with data as close to real data and with compute loads as close to real-world scenarios as possible. Have the traditional benchmark suites of HPC and database systems any value any more? What is the next step? These are the questions we aim to answer in this brief summary of benchmarking BigData systems.***

**INTRODUCTION.** Computable problems have driven scientists and engineers to build ever better, faster and more powerful computer systems. Along with increasing market interest came the need to prove how much better a system was than that of competitors. In the beginning the competition was primarily around database solutions, as they were the go-to for information storage, retrieval and compute resource in commercial settings. Competing vendors often made bold claims of their systems' performance in order to sell better. As often, however, they had little real-world relevance as the claims were based on vendor specific enumerations of their own systems. [7]

To alleviate the need for an objective metric that could compare systems fairly the Wisconsin benchmark suite was devised. This was a new way of computing, of measuring real-world performance, and gained much attention. The original research quoted on the measured performance of several state-of-the-art database systems. The research showed their relative performances which varied greatly, and consequently sparked a long standing benchmarking war. As a result, where once there were great differences in real-world performance, vendors needed to develop their systems to be able to compete and differences have therefore diminished. [7]

As traditional database systems have reached their limits regarding BigData, new systems – primarily MapReduce – have gained much popularity. It is to be expected, however, that database systems will continue to co-exist with new systems for the time being [23]. Unfortunately, along with traditional databases, the traditional benchmarks are also almost obsolete. We face a similar situation today, as did vendors and buyers of database systems some decades ago: there are no established standards in benchmarking BigData systems. Thus vendors may claim quasi-arbitrary properties of their systems. Without the standards – or metrics that come with standards – buyers have a difficult time in choosing the best system for their needs. Often clients have to buy into the system based on assumptions and only after having used the system for some time understand its capabilities and limitations. [11]

This seems to be especially the case with 'Cloud' [1] systems. On one hand it is fairly easy to try the services of a cloud provider, compared to the traditional setting of having to buy physical equipment. On the other hand there are many more cloud providers, less time, and more variation in services and their quality, than before to make informed decisions without objective metrics [10]. Fortunately the research and commercial communities have woken up to the situation and begun the development of benchmarks capable of testing BigData systems [1]. Today benchmarks will need to address several new problems, including the four V's of BigData and computation on it. This paper will outline important properties of the past benchmarks, review

[1]In this paper we refer to the more loose definition of a 'cloud': whether private or part of a 'public' enabled large computing system, a cloud is a large collection of interconnected computing resources accessible via the internet or otherwise, much like a grid-computing system.

upcoming systems and assess their relevance in the new era of BigData computing.

**BEFORE CLOUDS** were commonplace the compute resource consisted of a set of hardware and software written for that hardware. Hardware and its deployment was exceedingly expensive [10] and therefore buyers then, as much as today, needed guarantees of its suitability for their application domain. Commercial settings fueled the introduction of new database systems and before long the need for objective metrics was realised. In the following we outline some of the most popular benchmarks used in HPC and database settings and discuss their properties.

**The history of HPC-systems** benchmarking is perhaps longer than that of databases. It started around 1979 when the LINPACK-benchmark was introduced into the LINPACK-package. The aim of the benchmark was mainly to aid in the programming of systems designed to solve systems of linear equations by providing data on execution times. The appendix to the benchmark consisted of execution times on 23 popular computing systems of the era. LINPACK package contains the capability to perform many complex matrix operations using the subroutines of a lower-level BLAS package. The BLAS package is a collection of FORTRAN subroutines which make the core of the floating-point operations on matrices.

The package and benchmarks have expanded significantly since and encompass a wide array of calculations designed to evaluate the compute performance of a system. Currently the *Top500*[2] HPC ranking system uses the Highly-Parallel LINPACK (HPL) benchmark suite to measure system performance, demonstrating the evolution of the LINPACK package to measure the performance of new distributed memory and parallel systems. [9]

Since the beginning of benchmarking HPC systems and emergence of LINPACK, many benchmark suites have been introduced. Some of the most popular include, but are note limited to: SPEC CPU [8], HPCC [19] and PARSEC [2]. With perhaps the exception of PARSEC 2.0, they are benchmarks that concentrate mainly on hardware floating point operation speed. There is no notion of, for example, storage system benchmarking or network performance testing. As Wang, L. et al. note in their paper, these benchmarks are very low in their *operation intensity*, i.e. calculation versus memory addressing ratio [27]. Therefore they are not, by themselves, sufficient in measuring the overall performance of a computing system, especially when the system must perform other operations, such as I/O intensive data scanning, as well as raw number crunching. Such benchmarks can, however, show the road to testing the compute performance also in BigData scenarios and should not be dismissed as invalid. [11]

[2]http://www.top500.org

**The Wisconsin Benchmark** was devised in 1981 to satisfy a need for a benchmarking system with which to measure the performance of a specific DIRECT database machine. At the time there were no standard benchmarks. Only a few application specific benchmarks existed pertaining to the suitability of a system for that particular application. The newly developed benchmark received much attention perhaps primarily due to the fact that the publication contained the performance characteristics of real-world products and their names. The result was a benchmarking war and the Wisconsin Benchmark came to be known by vendors and customers alike. [7]

| RANK | SITE | SYSTEM | CORES | RMAX (TFLOP/S) | RPEAK (TFLOP/S) | POWER (KW) |
|---|---|---|---|---|---|---|
| 1 | National Super Computer Center in Guangzhou<br>China | **Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P<br>NUDT | 3,120,000 | 33,862.7 | 54,902.4 | 17,808 |
| 2 | DOE/SC/Oak Ridge National Laboratory<br>United States | **Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x<br>Cray Inc. | 560,640 | 17,590.0 | 27,112.5 | 8,209 |
| 3 | DOE/NNSA/LLNL<br>United States | **Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom<br>IBM | 1,572,864 | 17,173.2 | 20,132.7 | 7,890 |
| 4 | RIKEN Advanced Institute for Computational Science (AICS)<br>Japan | K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect<br>Fujitsu | 705,024 | 10,510.0 | 11,280.4 | 12,660 |
| 5 | DOE/SC/Argonne National Laboratory<br>United States | **Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom<br>IBM | 786,432 | 8,586.6 | 10,066.3 | 3,945 |
| 6 | DOE/NNSA/LANL/SNL<br>United States | **Trinity** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect<br>Cray Inc. | 301,056 | 8,100.9 | 11,078.9 | |
| 7 | Swiss National Supercomputing Centre (CSCS)<br>Switzerland | **Piz Daint** - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect , NVIDIA K20x<br>Cray Inc. | 115,984 | 6,271.0 | 7,788.9 | 2,325 |
| 8 | HLRS - Höchstleistungsrechenzentrum Stuttgart<br>Germany | **Hazel Hen** - Cray XC40, Xeon E5-2680v3 12C 2.5GHz, Aries interconnect<br>Cray Inc. | 185,088 | 5,640.2 | 7,403.5 | |
| 9 | King Abdullah University of Science and Technology<br>Saudi Arabia | **Shaheen II** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect<br>Cray Inc. | 196,608 | 5,537.0 | 7,235.2 | 2,834 |
| 10 | Texas Advanced Computing Center/Univ. of Texas<br>United States | **Stampede** - PowerEdge C8220, Xeon E5-2680 8C 2.700GHz, Infiniband FDR, Intel Xeon Phi SE10P<br>Dell | 462,462 | 5,168.1 | 8,520.1 | 4,510 |

**Figure 1: Current head Top500 HPC clusters[2].**

The Wisconsin Benchmark is well documented in [7] and will not be detailed here. It should be noted, however, that it was a flexible system designed to allow straightforward expansion – the specification of new relations and queries. Of further importance is that the benchmark introduced an industry first synthetic data generator that could be tuned to produce data with desired properties. Since the introduction of the Wisconsin Benchmark many other database evaluation systems have been developed.

The Transaction Processing Performance Council (TPC) is a non-profit organisation that focuses on designing benchmarks for system evaluation and disseminating objective results from the tested systems. For a long time the TPC-benchmarks have been regarded as industry standards for evaluating RDBMS system per-

formance. To understand why, we need to know and evaluate the TPC benchmarks with respect to the systems they were designed to benchmark. The current top-tier TPC[3] benchmarks consist of:.

**TPC-C:** On-line Transaction Processing (OLTP), simulates a population of users executing transactions against a database (1992).

**TPC-DI:** Data Integration: Export Transform Load (ETL) and OLTP type operations: manipulation of large volumes of various data and loading into a database (2013).

**TPC-DS:** Decision Support, simulates a retail product supplier, characterised by SQL queries, claims to provide a representative evaluation of the system under test also for BigData (2015).

**TPC-E:** OLTP, workload of a brokerage firm, database orientation (2015).

**TPC-H:** Ad-hoc DS benchmark, set of queries against a standard database (2013).

**TPC-VMS:** Leverages TPC-C, -E, -H and DS benchmarks to evaluate the performance of a virtualised database system (2013).

**TPCx-HS:** Evaluates the performance of BigData solutions of the Hadoop System variant (2015).

**TPCx-V:** A benchmark for virtual machine databases (2015).

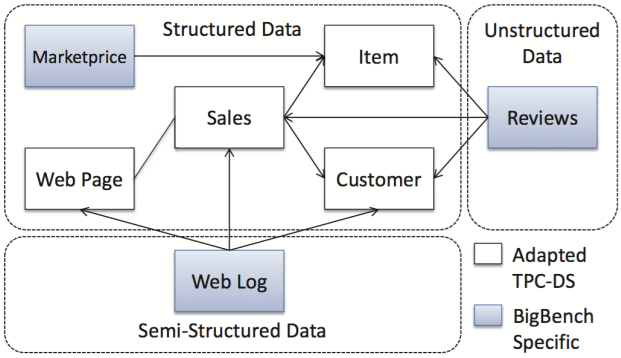


**Figure 2: The BigBench data model. Unstructured data is generated with a plug-in enhanced PDGF. [11]**

Most of the benchmarks concentrate on measuring the performance of a system revolving around a database. In addition to the current active benchmarks TPC has promoted the use of six, now obsolete, benchmarks. These obsolete benchmarks were also designed to test the transaction processing performance of various types of RDMBS in different scenarios. The variety of scenarios supported by current and obsolete TPC benchmarks can be seen as one of the primary reasons for their success, not to mention that TPC is non-profit.

Only recently has TPC introduced new benchmarks testing the handling of very large amounts of data. As can be seen, the TPC-C benchmark is still in active use demonstrating its relevance in benchmarking database systems. The TPC-DS and -HS, along with the upcoming TPCx-BB, demonstrate how the benchmarking industry is evolving in the advent of BigData.

## BENCHMARKING THE CLOUD

evidently requires new approaches since data sizes, CPU usages and network loads are no longer disconnected variables and system performance must be examined as a whole [27]. The following sections will cover recent developments in various problem scopes of BigData: data generation, workloads, and metrics. Further, a view into the future of benchmarking services will be presented including a discussion on work that has been conducted toward standardisation.

### Data used in benchmarking

must be made available to the system under test. How exactly this is achieved is a non-trivial problem as there are a multitude of usage scenarios, system environments, applications with different query strategies and so on, that need addressing. They all pose varying constraints and properties on the data.

One possibility could be to use real-world empirical data, but there are several problems with the approach. Firstly, if the data is to address 'Volume', then the dataset size would be prohibitively large to transfer across the internet from the facilities of one service provider to the other. Secondly, even though the data would have 'Variety', it would most probably be under privacy protection limiting its use or it is held by a corporation unwilling to share it for fear of exposing its competitive advantage. Thirdly, the dataset would not change unless some synthetic change was applied to it. 'Veracity' and 'Velocity of change' of the data could be addressed by a dataset that is a chronological change log, but again the size of the dataset would be ever more prohibitive. Thus using real-world data seems to be an unviable option. [24]

**PDGF.** For reasons stated the only option is to create the data synthetically. To that end, many data generators are in use in the industry[4] and are being developed in the academic world [3, 14]. In their paper authors T. Rabl et al. presented Parallel Data Generation Framework (PDGF), that is designed to tackle the difficulties of producing data for benchmarking cloud scale systems. It has been designed to be highly parallel and

[3]http://www.tpc.org/information/benchmarks.asp

[4]Turbo Data: http://www.turbodata.ca, GS Data Generator: http://www.GSApps.com

adaptable to diverse application and usage scenarios. [24]

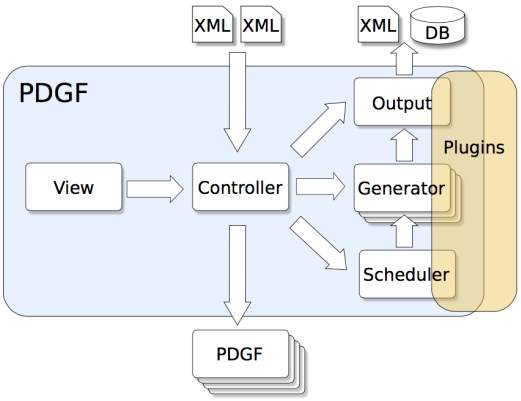


**Figure 3: A high level view of the PDGF [24].**

PDGF supports data generation in relational, Decomposed Storage Model, column, C-store and mixed modes. The data (see figure 3) to be generated is described in an XML configuration allowing flexibility in defining data properties. The approach the authors propose for data generation is that of a purely computational definition of attribute values. By mapping virtual row ID's to attribute values it becomes possible to reference a table entry of correlating data in any node of the system without the need to transfer the tuple across the network – the node that needs to reference said tuple can compute the tuple on-the-go. The functions to create uncorrelated and correlating data utilise parallel pseudo-random number generators and a deterministic seed to compute the attribute values, which can be done in any node regardless of the other nodes. This enables each data generating node to compute data independently without any network bottlenecks. PDGF was built to be easily extensible via plug-ins. The BigBench effort to build an industry standard BigData benchmark utilises PDGF as its data generator (figure 2) [11]. [24]

**PSDG.** Another similar data generator is the Parallel Synthetic Data Generator (PSDG) by Hoag, J. E. and Thompson, C. W.. It depends on the Synthetic Data Description Language (SDDL), also presented in the paper, to generate multi-terabyte datasets using computing clusters. SDDL, an XML-based language, is a flexible language for describing a wide variety of data. The flexibility of SDDL lies in its use of 'pools', that are hierarchically structured SDDL elements. Pools can contain auxiliary information, for instance names, zipcodes and their weights in a graph, that can be accessed during data generation. Pools, therefore, enable the user to define 'real' data amid synthetic data. Using pools the authors have modelled graph, map, state machine and context-free grammars in relational data format. Further description of the SDDL is beyond the scope of this paper and the reader is encouraged to read the referred white paper: [13].

High parallelism in PSDG is achieved by launching the PSDG data generating process on multiple machines in the cluster. Each process is responsible for creating a specific horizontal slice of the described data, and network communication is minimised by injecting the process with its parameters at start-up. PSDG lacks, however, the possibility to define complex statistical distributions often necessary in building complex data structures, such as synthetic scale-free graphs. Nevertheless PSDG is extensible via plug-ins which may be used to satisfy some deficiencies. [13]

Both PSDG and PDGF have been heavily influenced by the work of Gray, J. et al. on "Quickly generating billion-record synthetic databases" in [12]. The underlying principles in both data generators are relatively simple: minimise inter-process communication, provide an expressive language for data description, slicing the data deterministically to disjoint processes for high parallelism and making the parallelism transparent to the user.

**Workloads** differ widely from application to application, even within an application category such as MapReduce [4]. For example, scientific workloads are very heterogeneous and the system might bottleneck at the CPU, memory or I/O depending on application [21]. The same applies for serving applications [5]. In order to be able to provide a representative evaluation of a system, it is necessary that the benchmark support flexible programming models for expressive definition of usage scenarios [27]. Clouds are at the very cusp of the problem: hosting services for multiple customers inevitably results in multiple applications and thus various workloads [15]. Therefore, for example, no single TPC benchmark will suffice in evaluating holistic system performance.

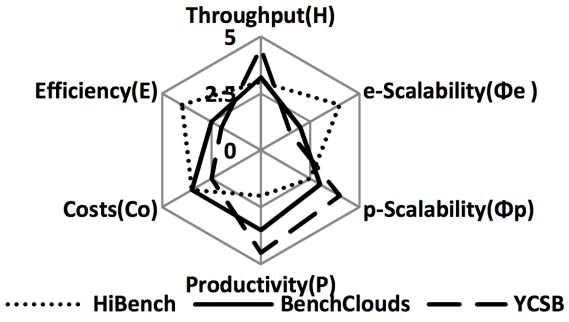


**Figure 4: A comparison of benchmark results. The YCSB indicates a higher scale-up elasticity than others [15].**

To illustrate, the authors of [15] used different benchmarks on Amazon EC2 Infrastructure as a Service (IaaS) evaluating its elastic scaling properties. Their results show, that where *Yahoo! Cloud Serving Benchmark* (YCSB) indicated a good 'scale-up' performance, it produced poorer results in evaluating 'scale-out' performance than HiBench and BenchClouds, the other two benchmarks assessed (figure 4). Among other findings they concluded, that the benchmarks they used in the evaluation produced diverging results [15].

Since the introduction of MapReduce, the programming paradigm has been widely adopted and is used in many application scenarios. Its simple, yet powerful semantics [17] have loaned themselves to those in need of business critical analysis, image processing and Monte-Carlo analysis, among others. These use cases present significantly different requirements on MapReduce deployments in their workloads. As pointed out by Chen Y. et al.[4], the earlier MapReduce benchmarks, e.g. GridMix, PigMix and Hive Benchmark, were designed to test the deployments using large datasets with representative workloads. These benchmarks, however, fall short of providing a true cross section of the various workloads MapReduce is employed in. To address this deficiency, authors Chen Y. et al. conducted research studying production traces to understand real MapReduce workloads and develop an ontology for describing them. They concluded that no single workload can cover the different usage scenarios of MapReduce. In lieu of using a single 'one size fits all' workload, they present workload suites and an associated synthetic workload generator, to cover various niches seen in production [4]. As an example, the TPCx-BB (BigBench) benchmark also details several use cases and associated workloads (see figure 5) further supporting the aforementioned claims.

Finally, *Yahoo!* designed their YCSB benchmark to evaluate the performance of several database systems for serving Web 2.0 [5]. It was further expanded by Patil S. et al. into YCSB++ to evaluate advanced features in table stores [22]. Other work in the field include that of Sobel W. et al. also concentrating on building a set of tools to evaluate the end-to-end performance of cloud serving systems built on IaaS like that of Amazon EC2 [25].

**Metrics** define how we perceive the results of a benchmark. In HPC communities it has been typical to report system performance as the number of floating point operations per second, FLOPS [9]. The traditional database benchmarks have reported the transactions per second (TPS) that the evaluated system was capable of [7]. Although Ghazal, A. et al. provide a counter argument [11], in the context of BigData and cloud systems it is no longer sufficient to report system performance with a single number. Rather, multiple dimensions must be analysed and reported for a fair representation of a system. Given the set of Key Performance Indicators (KPI), users can make informed decisions about the systems and choose the one that best suits their needs [10]. In addition, performance studies may reveal problem areas to the service providers, enabling further development [18].

| Query processing type | Total | Percentage(%) |
|---|---|---|
| Declarative | 10 | 33.3 |
| Procedural | 7 | 23.3 |
| Mix of Declarative and Procedural | 13 | 43.3 |
| **Data sources** | **Total** | **Percentage(%)** |
| Structured | 18 | 60.0 |
| Semi-structured | 7 | 23.3 |
| Un-structured | 5 | 16.7 |
| **Analytic techniques** | **Total** | **Percentage(%)** |
| Statistics analysis | 6 | 20.0 |
| Data mining | 17 | 56.7 |
| Reporting | 8 | 26.7 |

**Figure 5: Technical breakdown of BigBench workloads[6]. [11]**

Among others [18, 15, 26], the authors of CSMIC [10] outline a broad range of KPIs that form a representative cross section of system performance and quality of service (QoS). The chosen high-level QoS attributes are based on ISO standards: Accountability, Agility, Assurance of Service, Cost, Performance, Security and Privacy, and Usability. The following decomposition summarises the proposed KPIs to measure clouds [10]:

**Response time:** how fast a service (e.g. virtual machine) is available.

**Sustainability:** how environmentally sustainable the service is (e.g. carbon footprint).

**Suitability:** how well user requirements are met (essential and non-essential functions covered).

**Accuracy:** how close the measured aspects of the service are to expectations (e.g. SLA violations).

**Transparency:** how transparent (unnoticeable) changes in the service are to users (e.g. major API changes).

**Interoperability:** how well the service integrates with other services (e.g. external data sources).

**Availability:** how often the user is able to access the service.

**Reliability:** how often failures occur in the service (less failures means more reliable).

**Stability:** how much does the service performance vary (e.g. I/O speed fluctuation).

[6]Processing type refers to the type of language needed to answer the query: SQL for declarative, MapReduce for procedural. Data sources refers to the type of data addressed in the query. Analytic techniques contain statistics (e.g. regression) or simple reporting (e.g. number of sold items) and complex data mining (e.g. association mining).

**Cost:** how much a service costs to purchase and maintain (mostly application specific).

**Adaptability:** how well a service provider can adapt to user requests (e.g. VM upgrades).

**Elasticity:** how well the service scales on load (adding or removing VM instances dynamically).

**Usability:** how easy it is to use the service (e.g. operability and installability).

**Throughput and efficiency:** how many tasks the service is able to complete in a given time.

**Scalability:** how well resources can be provided to manage requests (vertical scalability).

Each of the aforementioned KPIs may belong to one or multiple top-level QoS categories. Accompanying the QoS and KPI definitions, the authors present a ranking system based on an Analytical Hierarchical Process mechanism. The ranking process is influenced by measured statistics of the service along with user preferences, as noted in the summary of the KPIs. While the SMICloud framework was developed specifically for ranking service providers and thus provides no low-level benchmarks, it can make use of their results [10].

As can be seen the list of KPIs is quite exhaustive and covers far more metrics than just the aspect of compute or storage performance. Should a customer choose to use all of the KPIs is their choice, but at the very least they have that option. An option many benchmarks don't necessarily offer at all.

**INDUSTRY STANDARDS** do not exist as of yet. Mainly due to the fact, that field of benchmarking BigData systems is still very young. The industry has not yet had the time to mature with the current propositions for benchmarking standards. There may be a strong pull towards existing leaders, for example TPC with the BigBench-benchmark [11]. Having already been adapted by TPC in their upcoming TPCx-BB[7] benchmark suite, BigBench to be gaining traction. Of further interest may be the TPCx-HS benchmark aimed to be an industry standard in benchmarking Hadoop based MapReduce systems [20].

However, more than ever we are seeing the emergence of Benchmarking as a Service. Increasing interest in cloud systems has brought this new market segment to life. Here, third party organisations and companies[8] invest solely into evaluating existing cloud providers' services and sell the results to potential – or existing, for that matter – cloud customers. Even some cloud providers offer benchmarking services and may publish some general results for free, then sell customised services[9].

**Scientific computing** in the cloud is an interesting domain that may arise from benchmark results. Scientific computing requires enormous resources which often are scarce due to shortages in government funding for universities. With the pay-as-you-go model of the cloud it may become a viable option particularly for small research groups to buy compute power from a cloud provider [16]. That is, if the services offer good compute power at a reasonable price – this is where benchmarking comes into play.

In the past performance evaluations have been conducted in this context. In 2010 the results were not so promising [16], but since then new services have been introduced. For example the Amazon Cluster Compute is aimed at HPC customers. It may prove to provide the resources and provisioning models favourable to scientific computing [21], but this remains to be seen.

**CONCLUDING REMARKS.** With the traditional database and HPC era coming to a close, the era of the Cloud and BigData is well on its way. The disillusionment about cloud capabilities is stepping in and we are starting their objective quantification. Benchmarks are being developed for that purpose, just as they were developed for databases and HPC systems. Much research has and is being poured into understanding the complexities of benchmarking BigData systems. New properties, such as elastic scaling, virtualised databases, and software defined networks, need new evaluation tools. New storage models, such as graph storages [6], need new benchmarks as well.

BigData benchmarking needs new tools and methods in generating the data that the benchmark uses. Much can be learned from past examples; empirical real world data is not an option due in part to the volume of the data needed. PDGF and PSDG seem like promising candidates to set the stage for new data generation approaches with their massively parallel and flexible data models. It will be seen, however, how easily extensible the tools are, because by themselves they can satisfy only a very limited number of uses. PDGF, having been incorporated into the BigBench and TPCx-BB benchmarks has a slight lead in this respect.

The industrial and scientific communities have shown that clouds, clusters, private computing centres and the like have many uses. With many uses comes the inevitable burden of a plethora of workload types. As a result, benchmarking such systems with one type of workload does not provide a representative overview of the system under test. We have seen that HPC-style benchmarks cover only a very small slice of the work-

[7] http://www.tpc.org/tpcx-bb/default.asp

[8] for example: https://cloudharmony.com, http://cloudspectator.com, http://serverbear.com

[9] https://www.profitbricks.com/cloud-performance-testing/

load types, namely computational. Database benchmarks, such as the renowned TPC-C, focused on the transaction processing performance and little attention was given to computational aspects, such as machine learning. Systems must, therefore, be tested with as many different types of workloads and benchmarks fulfilling this need must be devised.

There are many methods of presenting the results of a benchmark. It should be evident, that the more aspects are reported, the more insight any single party may gain. Compressing the results of a multi-faceted benchmark into a single number inevitably causes information loss and even substantial skewing of the result – for example in a case where one KPI is in a different order of magnitude. If the result vector can be processed with weights then the relevance of the weights needs to be assessed by the party looking to use the results. Should the weights be user defined, the end-user may have more use of the benchmarking report due to the fact that each party has different use cases and therefore different needs for the system. In the end it is the users' needs that dictate how useful any single benchmark and result is.

## 1. REFERENCES


[1] Aversano, G., Rak, M., and Villano, U. The mosaic benchmarking framework: Development and execution of custom cloud benchmarks. *Scalable Computing: Practice and Experience 14*, 1 (2013), 33–46.

[2] Bienia, C. *Benchmarking Modern Multiprocessors*. PhD thesis, Princeton University, January 2011.

[3] Bruno, N., and Chaudhuri, S. Flexible database generators. In *Proceedings of the 31st international conference on Very large data bases* (2005), VLDB Endowment, pp. 1097–1107.

[4] Chen, Y., Ganapathi, A., Griffith, R., and Katz, R. 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* (2011), IEEE, pp. 390–399.

[5] Cooper, B. F., Silberstein, A., Tam, E., Ramakrishnan, R., and Sears, R. Benchmarking cloud serving systems with ycsb. In *Proceedings of the 1st ACM symposium on Cloud computing* (2010), ACM, pp. 143–154.

[6] Dayarathna, M., and Suzumura, T. Xgdbench: A benchmarking platform for graph stores in exascale clouds. In *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on* (2012), IEEE, pp. 363–370.

[7] DeWitt, D. J. The wisconsin benchmark: Past, present, and future., 1993.

[8] Dixit, K. M. Overview of the spec benchmarks., 1993.

[9] Dongarra, J. J., Luszczek, P., and Petitet, A. The linpack benchmark: past, present and future. *Concurrency and Computation: practice and experience 15*, 9 (2003), 803–820.

[10] Garg, S. K., Versteeg, S., and Buyya, R. A framework for ranking of cloud computing services. *Future Generation Computer Systems 29*, 4 (2013), 1012–1023.

[11] Ghazal, A., Rabl, T., Hu, M., Raab, F., Poess, M., Crolotte, A., and Jacobsen, H.-A. 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, pp. 1197–1208.

[12] Gray, J., Sundaresan, P., Englert, S., Baclawski, K., and Weinberger, P. J. Quickly generating billion-record synthetic databases. In *ACM SIGMOD Record* (1994), vol. 23, ACM, pp. 243–252.

[13] Hoag, J. E., and Thompson, C. W. A parallel general-purpose synthetic data generator. *ACM SIGMOD Record 36*, 1 (2007), 19–24.

[14] Houkjær, K., Torp, K., and Wind, R. Simple and realistic data generation. In *Proceedings of the 32nd international conference on Very large data bases* (2006), VLDB Endowment, pp. 1243–1246.

[15] Hwang, K., Bai, X., Shi, Y., Li, M., Chen, W.-G., and Wu, Y. Cloud performance modeling with benchmark evaluation of elastic scaling strategies. *Parallel and Distributed Systems, IEEE Transactions on 27*, 1 (2016), 130–143.

[16] Iosup, A., Ostermann, S., Yigitbasi, M. N., Prodan, R., Fahringer, T., and Epema, D. H. Performance analysis of cloud computing services for many-tasks scientific computing. *Parallel and Distributed Systems, IEEE Transactions on 22*, 6 (2011), 931–945.

[17] Jiang, D., Ooi, B. C., Shi, L., and Wu, S. The performance of mapreduce: An in-depth study. *Proceedings of the VLDB Endowment 3*, 1-2 (2010), 472–483.

[18] Li, A., Yang, X., Kandula, S., and Zhang, M. Cloudcmp: comparing public cloud providers. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (2010), ACM, pp. 1–14.

[19] Luszczek, P. R., Bailey, D. H., Dongarra, J. J., Kepner, J., Lucas, R. F., Rabenseifner, R., and Takahashi, D. The hpc challenge (hpcc) benchmark suite. In *Proceedings of the 2006 ACM/IEEE conference on Supercomputing* (2006), Citeseer, p. 213.

[20] Nambiar, R. Benchmarking big data systems: Introducing tpc express benchmark hs. In *Big Data Benchmarking*. Springer, 2014, pp. 24–28.

[21] Ostermann, S., Iosup, A., Yigitbasi, N., Prodan, R., Fahringer, T., and Epema, D. A performance analysis of ec2 cloud computing services for scientific computing. In *Cloud computing*. Springer, 2009, pp. 115–131.

[22] Patil, S., Polte, M., Ren, K., Tantisiriroj, W., Xiao, L., López, J., Gibson, G., Fuchs, A., and Rinaldi, B. Ycsb++: benchmarking and performance debugging advanced features in scalable table stores. In *Proceedings of the 2nd ACM Symposium on Cloud Computing* (2011), ACM, p. 9.

[23] Pavlo, A., Paulson, E., Rasin, A., Abadi, D. J., DeWitt, D. J., Madden, S., and Stonebraker, M. A comparison of approaches to large-scale data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data* (2009), ACM, pp. 165–178.

[24] Rabl, T., Frank, M., Sergieh, H. M., and Kosch, H. A data generator for cloud-scale benchmarking. In *Performance Evaluation, Measurement and Characterization of Complex Systems*. Springer, 2010, pp. 41–56.

[25] Sobel, W., Subramanyam, S., Sucharitakul, A., Nguyen, J., Wong, H., Klepchukov, A., Patil, S., Fox, A., and Patterson, D. Cloudstone: Multi-platform, multi-language benchmark and measurement tools for web 2.0. In *Proc. of CCA* (2008), vol. 8.

[26] Turner, A., Fox, A., Payne, J., and Kim, H. S. C-mart: Benchmarking the cloud. *Parallel and Distributed Systems, IEEE Transactions on 24*, 6 (2013), 1256–1266.

[27] Wang, L., Zhan, J., Luo, C., Zhu, Y., Yang, Q., He, Y., Gao, W., Jia, Z., Shi, Y., Zhang, S., et al. Bigdatabench: A big data benchmark suite from internet services. In *High Performance Computer Architecture (HPCA), 2014 IEEE 20th International Symposium on* (2014), IEEE, pp. 488–499.