**Understanding Big Data Applications and Architectures**

*Geoffrey Fox, Judy Qiu, Shantenu Jha, Indiana and Rutgers University*

We have studied the capabilities of many Apache Big Data software projects contributing to what we can term ABDS or Apache Big Data Stack and developed a layered architecture with approximately 120 software subsystems with over 40 from Apache. We suggest this architecture can be used to understand the key abstractions and standards needed for Big Data. Further in a separate abstract, we propose integrating key capabilities from HPC and ABDS to lead to a high performance Big Data stack. This needs attention to abstractions and features like a) Messaging and Collective/reduction operations from MPI (HARP), BSP, Hadoop, Giraph; b) External Data Interface including basic HDFS; c) Fault Tolerance; d) Caching for iterative jobs; e) Choice of internal data format compatible with in-memory databases or simple arrays (probably special case of in-memory data arrays); f) SPMD parallelism; g) Serialization for efficient communication; h) Hybrid runtime with threads and/or processes and i) Application level scheduling.

We study this architecture in the context of a classification of representative applications called ogres that have three facets or ways of organizing their structure. This draws heavily on the use cases collected in NIST process last year as described in a separate abstract.

The first facet captures different problem architectures. Some representative examples are (i) Pleasingly Parallel – as in Blast (over sequences), Protein docking (over proteins and docking sites), imagery, (ii) Local Machine Learning (ML) – or filtering pleasingly parallel as in bio-imagery, radar (this contrasts with Global Machine Learning seen in LDA, Clustering etc. with parallel ML over nodes of system), (iii) Fusion – where knowledge discovery often involves fusion of multiple methods (ensemble methods are one approach), (iv) Data points in metric or non-metric spaces, (v) Maximum Likelihood, (vi) ­ 2 minimizations, (vii) Expectation Maximization (often Steepest descent), and (viii) Quantitative measures for Big Data applications which can be captured by absolute sizes and relative ratios of flops, IO bytes and communication bytes.

The second facet captures applications with important data sources with distinctive features, representative examples of the data sources include, (i) SQL based, (ii) NOSQL based, (iii) Other enterprise data systems, (iv) Set of Files (as managed in iRODS), (v) Internet of Things, (vi) Streaming, (vii) HPC simulations, and (viii) Temporal features – for in addition to the system issues, there is a temporal element before data gets to compute system, e.g., there is often an initial data gathering phase which is characterized by a block size and timing. Block size varies from month (Remote Sensing, Seismic) to a day (genomic) to seconds (Real time control, streaming)

The third facet contains Ogres themselves classifying core analytics and kernels/mini-applications/skeletons, with representative examples (i) Recommender Systems (Collaborative Filtering) (ii) SVM and Linear Classifiers (Bayes, Random Forests), (iii) Outlier Detection (iORCA) (iv) Clustering (many methods), (v) PageRank, (vi) LDA (Latent Dirichlet Allocation), (vii) PLSI (Probabilistic Latent Semantic Indexing), (viii) SVD (Singular Value Decomposition), (ix) MDS (Multidimensional Scaling), (x) Graph Algorithms (seen in neural nets, search of RDF Triple stores), (xi) Neural Networks (Deep Learning), (xii) Global Optimization (Variational Bayes), (xiii) Agents, as in epidemiology (swarm approaches) and (xiv) GIS (Geographical Information Systems)