Contributions to High-Performance Big Data Computing

Geoffrey FOX [a,1], Judy QIU [a], David CRANDALL [a], Gregor VON LASZEWSKI [a], Oliver BECKSTEIN [b], John PADEN [c], Ioannis PARASKEVAKOS [d], Shantenu JHA [d], Fusheng WANG [e], Madhav MARATHE [f], Anil VULLIKANTI [f] and Thomas CHEATHAM [g]

[a] *Indiana University, gcf@iu.edu, xqiu@iu.edu, djcran@indiana.edu, gvonlasz@iu.edu,*
[b] *Arizona State University , obeckste@asu.edu,*
[c] *Kansas University, paden@ku.edu,*
[d] *Rutgers University, shantenu.jha@rutgers.edu, ioannis.paraskevakos@rutgers.edu*
[e] *Stony Brook University, fusheng.wang@stonybrook.edu,*
[f] *Virginia Tech (now at the University of Virginia) mvm7hz@virginia.edu, vsakumar@virginia.edu,*
[g] *University of Utah, tec3@utah.edu*

Abstract. Our project is at the interface of Big Data and HPC -- High-Performance Big Data computing and this paper describes a collaboration between 7 collaborating Universities at Arizona State, Indiana (lead), Kansas, Rutgers, Stony Brook, Virginia Tech, and Utah. It addresses the intersection of High-performance and Big Data computing with several different application areas or communities driving the requirements for software systems and algorithms. We describe the base architecture, including the HPC-ABDS, High-Performance Computing enhanced Apache Big Data Stack, and an application use case study identifying key features that determine software and algorithm requirements. We summarize middleware including Harp-DAAL collective communication layer, Twister2 Big Data toolkit, and pilot jobs. Then we present the SPIDAL Scalable Parallel Interoperable Data Analytics Library and our work for it in core machine-learning, image processing and the application communities, Network science, Polar Science, Biomolecular Simulations, Pathology, and Spatial systems. We describe basic algorithms and their integration in end-to-end use cases.

Keywords. HPC, Big Data, Clouds, Graph Analytics, Polar Science, Pathology, Biomolecular simulations, Network Science, MIDAS, SPIDAL

1. INTRODUCTION AND MOTIVATION

1.1. Activities

This work is a collaboration between university teams at Arizona State, Indiana (lead), Kansas, Rutgers, Stony Brook, Virginia Tech, and Utah. Our website is [1] describing work funded by the US National Science Foundation as NSF 1443054. The project is

[1]Corresponding Author, Digital Science Center, Indiana University Bloomington, IN, USA. E-mail: gcf@iu.edu

constructing data building blocks to address major cyberinfrastructure challenges in several different communities including Biomolecular Simulations, Network, and Computational Social Science, Computer Vision, Spatial Geographical Information Systems, Remote Sensing for Polar Science, Pathology Informatics as well as core machine learning and optimization techniques. We are now exploring other possible communities such as those using streaming data systems where our work could be valuable. This paper updates an extensive project report from Fall 2016 [2] and extends a recent poster [3].

One set of the project building blocks are the components of our **Middleware for Data-Intensive Analytics and Science (MIDAS)** built by Indiana University and Rutgers that enables scalable applications with the performance of HPC (High-Performance Computing) and the rich functionality of the commodity Apache Big Data Stack. The other set of building blocks are the currently around 40 parallel high-performance members (see sections 4.2 and 4.3) of our **Scalable Parallel Interoperable Data Analytics Library (SPIDAL)**. Interoperability implies a broad target for our software, which generates what we call HPC clouds on XSEDE systems, public clouds and petascale to exascale supercomputers. We support Java or applications written in more traditional HPC languages like C++.

The project has designed and is using an overall architecture built around the twin concepts of **High-Performance Computing enhanced Apache Big Data Stack (HPC-ABDS)** software and classification of Big data applications – the Ogres – that defined the key qualities exhibited by applications and required to be supported in software. This key early work is well published and mature enough to guide our software architecture and choice of broadly usable building blocks. It has had a significant impact on the NIST Big Data Interoperability Framework [4] and is summarized in section 2. We have also proposed a Big Data – Big Simulation and HPC convergence based on these ideas, which suggest that our SPIDAL library and MIDAS middleware will be important for realizing convergence. We have been working closely with the BDEC (Big Data Exascale Computing) initiative on both convergence and more broadly use of HPC in Big Data systems [5]. The kickoff BDEC2 meeting was hosted by Indiana University in November.

MIDAS mixes traditional HPC technologies such as MPI, Pilot Jobs, and Slurm with Big Data environments such as Hadoop, Spark, Flink, Heron, Storm, Docker, Mesos, Kubernetes, and Tensorflow. We have several published studies that look at partial integrations such as MPI and Hadoop or Slurm and Heron. However, now we have designed a clean HPC-Big Data middleware with 3 key components: **Pilot Jobs**, Harp, and Twister2. Pilot jobs have successfully been extended from simulations to Big Data. Harp integrates with Hadoop providing both collective communication and the Intel Data Analytics node library DAAL (Data Analytics Acceleration Library). The success of the Intel Parallel Computing Center@IU is built around this. Finally, Twister2 aims to provide the full functionality of Spark Flink and Heron including dataflow communication (where we have released a library Twister: Net), RDD-style in-memory databases, SQL modules but with integrated high-performance mechanisms and software.

1.2. Big Data Application Analysis; Big Data Ogres

A major achievement of the project was the development of the Big Data Ogres and their use to describe the integration of HPC, Big Data, and Simulations. In this

completed work, the Big Data Ogres built on a collection of 51 big data uses gathered by the NIST Public Working Group where 26 properties were gathered for each application [4]. This information was combined with other studies including the Berkeley dwarfs [6], the NAS parallel benchmarks [7], [8] and the Computational Giants of the NRC Massive Data Analysis Report [9]. The Ogre analysis led to a set of 50 features divided into four views that could be used to categorize and distinguish between applications [10]. The four views are Problem Architecture (Macropattern); Execution Features (Micropatterns); Data Source and Style; and finally the Processing View or runtime features. We generalized [11] this approach to integrate Big Data and Simulation applications into a single classification that we called convergence diamonds with the total facets growing to 64 in number and split between the same 4 views and this is shown in fig. 1-1. Further, a mapping of facets into the work of this project has been given earlier [10].

1.3. Global Artificial Intelligence and Modelling Supercomputer

Recently Microsoft research described their computing systems work under the umbrella term of the Global AI Supercomputer. We can expand this idea to describe our converged big data and big simulation approach as a Global Artificial Intelligence and Modelling Supercomputer (GAIMS). We add modeling to include simulations and to recognize that big data analysis includes both data and model. GAIMS consists of an intelligent cloud integrated with an intelligent edge. The intelligent cloud is logically centralized but physically distributed consisting by 2021 of over 600 hyperscale data centers (roughly over 50,000 servers each). There will be by 2021 around 14 billion links to the Intelligent Edge or IoT, which is a distributed Data Grid holding over 80% of the digital data created each year [12], [13]. Both the Intelligent Cloud and Intelligent Edge form Grids in a parlance that has recently fallen out of favor. Compared to previous work on Grids, today's realization has a much greater emphasis on data and as used has a clearer administrative structure.

Edge devices and local fog computing offer much better latency on limited computing tasks with single tenancy but security risks from the nonprotected deployment and often immature software systems. In our project, we are using High Performance Computing ideas/technologies to give better functionality and performance "cloud" and "edge" systems. The edge for scientific research can involve devices from giant accelerators and light sources, remote sensing satellites, gene sequences to tiny environmental sensors driving AI First Science and Engineering. GAIMS delivers digital twin simulations to industry and computational science simulations to academia. It analyses research, government, commercial, and consumer data. GAIMS realizes the goals of High-Performance Big-Data Computing (HPBDC), Big Data and Extreme-scale Computing (BDEC), and the Common Digital Continuum Platform for Big Data and Extreme Scale Computing (BDEC2). Below section 2 describes the technology to power GAIMS and sections 3 and 4 its applications. In the final section 6, we comment that GAIMS both powers AI First activities but is itself enhanced by AI configuring and learning its behavior. [14]–[17]

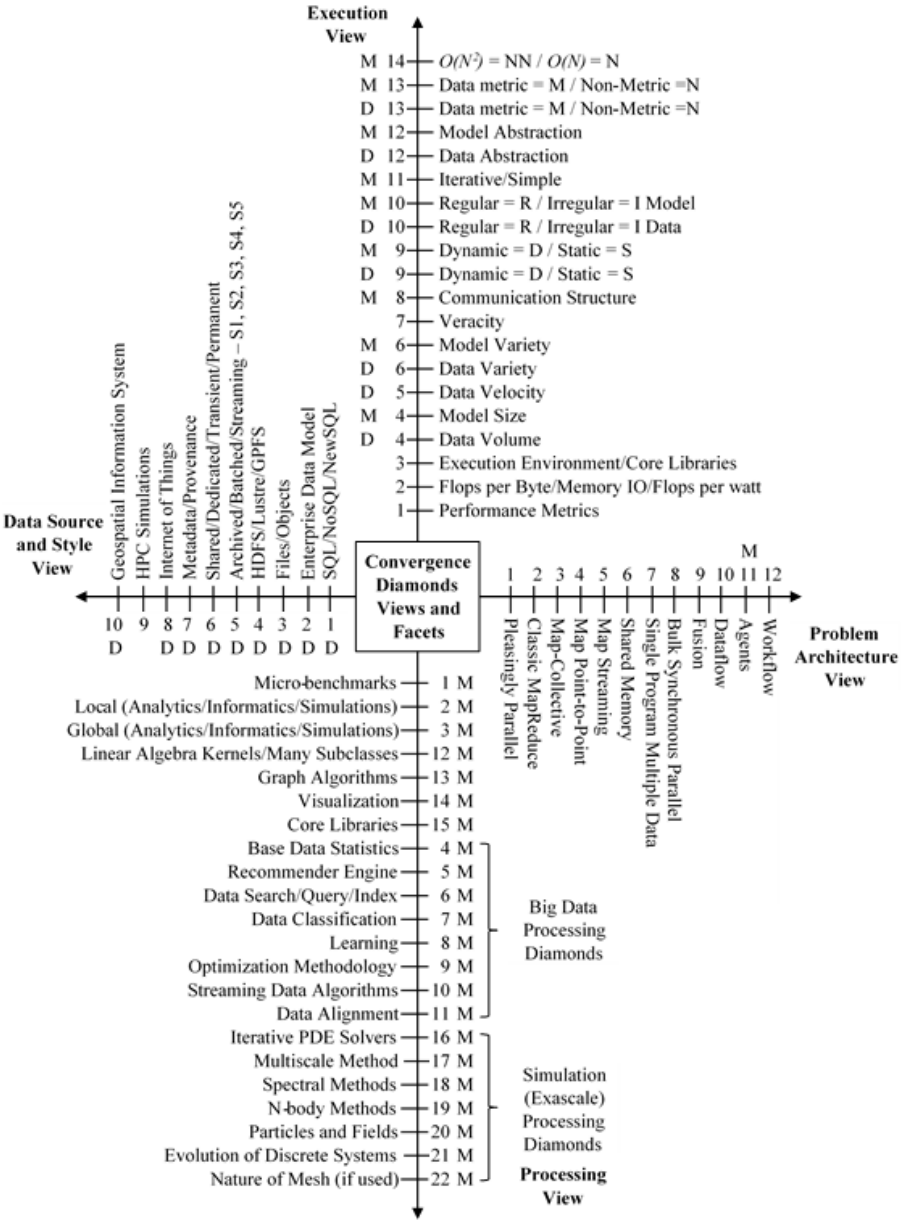

Figure 1-1; 64 Convergence Diamonds [11] Building on Big Data Ogres

2. MIDDLEWARE BUILDING BLOCKS (MIDAS)

2.1. High-Performance Computing enhanced Apache Big Data Stack

An important but completed project gathered in Figure 2-1 much existing relevant systems software coming from either HPC or commodity Apache or other Big Data sources [18]. We stopped collecting software names as although the active field was still generating new software names, we felt the message and even software architecture was clear, and we did not gain anymore from increasing our collection which started in 2013 with just 35 members [19].

The software is broken up into 21 layers so software systems are grouped by functionality and the layers where there is special attention in this project are colored green in Figure 2-1. This software collection is termed HPC-ABDS (High Performance Computing enhanced Apache Big Data Stack) as many critical core components of the commodity stack (such as Hadoop, Spark, and HBase) come from open source projects while HPC is needed to bring performance and other parallel computing capabilities as we described in sections 1.3 and 4.1 [20].

Note that Apache is the largest but not the only source of open source software; we believe that the Apache Foundation is a critical leader in the Big Data open source software movement and use it to designate the full big data software ecosystem. The figure also includes proprietary systems as they illustrate key capabilities and often motivate open source equivalents. We built this picture for big data problems, but it also applies to big simulation with the caveat that we need to add more high-level software at the library level and more high-level tools like Global Arrays.

One essential idea of our Big Data HPC convergence for software is to make use of ABDS software where possible as it offers richness in functionality, a compelling open-source community sustainability model and typically attractive user interfaces. ABDS has a good reputation for scale but often does not give good performance. Our approach augments ABDS with HPC ideas, where we illustrated this with Harp for Hadoop [21]–[23], Storm [24], and the basic Java environment [25]. Twister2 [26], [27] in section 2.4 builds a toolkit from the best of Spark, Heron (Storm), Kafka, Harp-DAAL, MPI, Mesos, Slurm and Kubernetes and offers an HPC-ABDS Platform as a Service. We suggest that GAIMS can use HPC-ABDS for both big data and big simulation applications.

2.2. Harp-DAAL Computation Model and Collective Communication

The convergence of high performance computing, big data, and machine learning will enable new software capabilities that seamlessly incorporate simulation and data analytics. Our research categorizes data-intensive computation for big data, and big compute applications into five computation models (or abstractions) that map into five distinct system architectures [10]. It starts with Sequential, followed by centralized batch architectures corresponding exactly to the three forms of MapReduce: Map-Only, MapReduce and Iterative MapReduce. Category five is the classic MPI model. We introduced a novel HPC-Cloud convergence framework named Harp-DAAL [28]–[31]. Harp-DAAL shows how simulations and Big Data can use common programming environments[11] with a runtime based on a rich set of collectives and libraries.

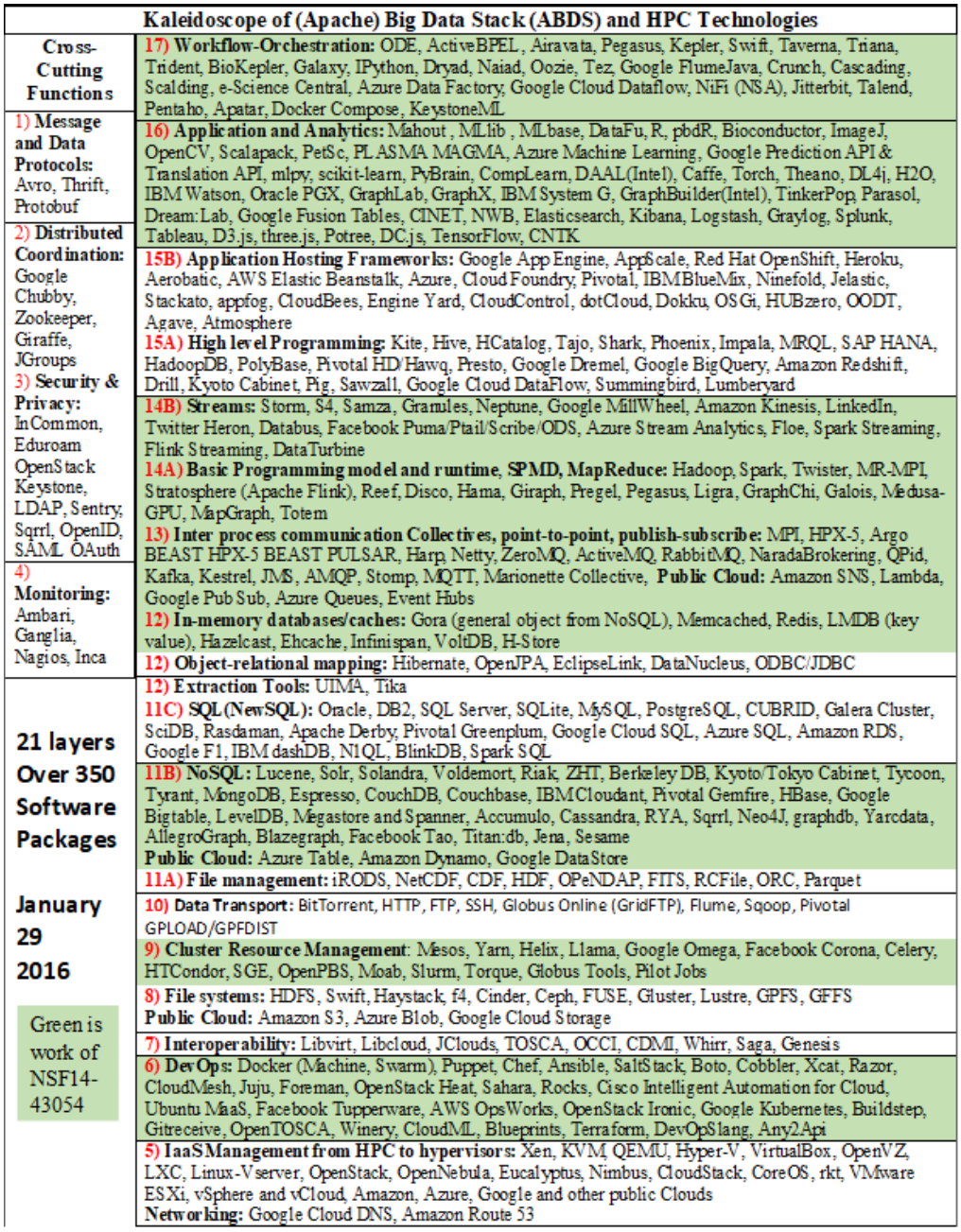

Figure 2-1: HPC-ABDS as compiled January 29, 2016, with layers of particular interest in this project shown in green [18].

Harp is a collective communication library that supports all 5 classes of data-intensive computation, from pleasingly parallel to machine learning and simulations. Harp expanded the applicability of Hadoop (with Harp plugin) for more classes of Big Data applications, especially complex data analytics such as machine learning and graph. Harp uses Intel® Data Analytics Accelerator Library (DAAL) [32], for its highly

optimized kernels on Intel® Xeon and Xeon Phi architectures. This way, the high-level API of Big Data tools can be combined with intra-node fine-grained parallelism that is optimized for HPC platforms. The current Harp-DAAL repository discussed in sections 4.2 and 4.3 includes over 40 batch and distributed modes of algorithms in Intel DAAL 2019 version. Recent work can be found in [33], [34] as well as a thorough report [28] and SC2017 tutorial[35].

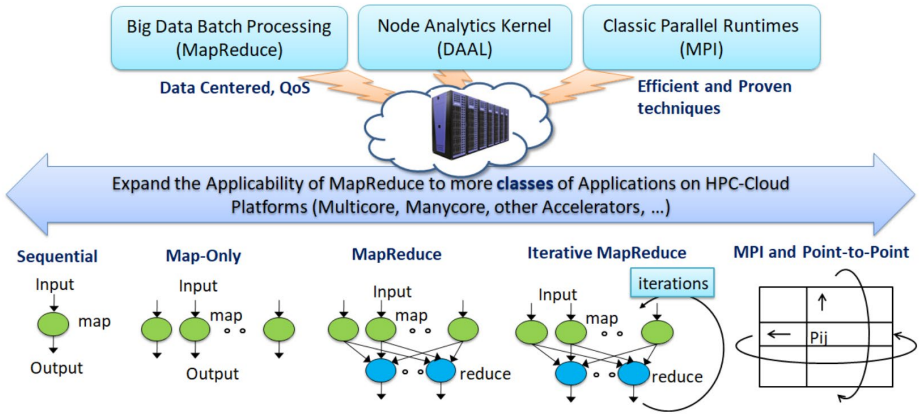

Figure 2-2 Cloud-HPC interoperable software for High Performance Big Data Analytics at Scale

2.3. RADICAL-Cybertools

The Pilot abstraction [36] has been used in HPC environments for supporting a diverse set of task-based workloads. A Pilot-job is a placeholder job which is submitted to the resource manager of an HPC and acquires a set of resources where an arbitrary number of compute tasks can be executed. This functionality allows all the computational tasks to be executed directly on the resources, without being queued again. The pilot abstraction thus supports the requirements of task-level parallelism and high throughput as needed by science drivers, without affecting or circumventing the queue policies of HPC resources.

RADICAL-Pilot (RP) [37] is an implementation of the pilot abstraction, engineered to support scalable and efficient execution of heterogeneous tasks across different platforms. Workloads and pilots are described via the Pilot API and passed to RP's runtime system. The PilotManager submits pilots as jobs (or virtual machines or containers) to one or more Cyber-Infrastructures (CIs) via the SAGA API. The SAGA API implements an adapter for each supported type of CI, exposing uniform methods for job and data management. Once a pilot becomes active on a CI, it bootstraps the Agent module. The UnitManager schedules each task to an Agent via a queue on a MongoDB instance. Each Agent pulls its tasks from the DB module scheduling them on the Executor. The Executor sets up the task's execution environment and then spawns the task for execution. When required, the input data of a task are either pushed to the Agent or pulled from the Agent, depending on data locality and sharing requirements. Similarly, the output data of the task are staged out by the Agent and UnitManager to a specified destination, e.g., a filesystem accessible by the Agent or the user workstation. Both input and output staging are optional, depending on the requirements of the tasks. The actual file transfers are enacted via SAGA, and currently

support (gsi)-scp, (gsi)-sftp, Globus Online, and local and shared file system operations via cp. Consequently, the size of the data along with network bandwidth and latency or filesystem performance determine the data staging durations.

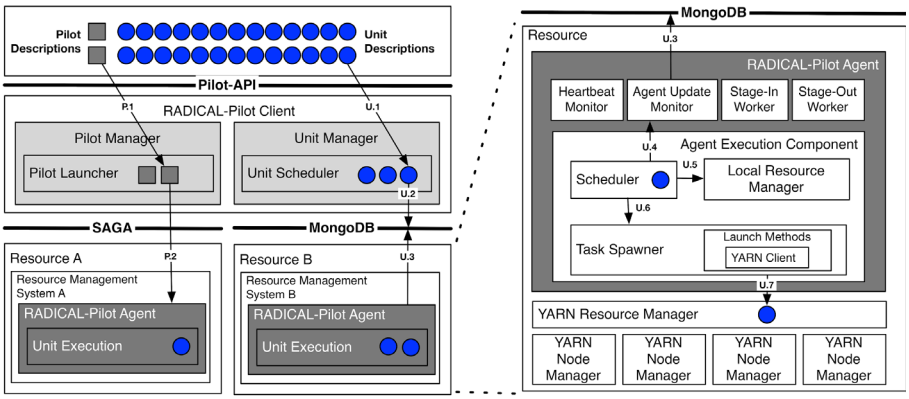

Figure 2-3: RADICAL-Pilot and Pilot-Hadoop integration

RADICAL-Pilot has been extended to support big data processing and streaming jobs by utilizing Hadoop Spark and Kafka. **Pilot-Hadoop** [38] (Fig 2-3) enables the dynamic, ad-hoc creation of Hadoop or Spark clusters on HPC infrastructures. It allows users to execute scalable analytics, such as iterative machine learning, without the need to manage a Hadoop or Spark cluster on HPC. Pilot-Hadoop can be used via the same API as RADICAL-Pilot. As a result, a common API is provided for executing different types of workloads, e.g., MPI, YARN, or Spark. **Pilot-Streaming** [39], is a framework for supporting streaming frameworks, applications, and their resource management needs on HPC infrastructures. Pilot-Streaming is based on the Pilot-Job concept and enables developers to manage distributed computing and data resources for complex streaming applications. It enables applications to dynamically respond to resource requirements by adding/removing resources at runtime. This capability is critical for balancing complex streaming pipelines

2.4. Twister2

Twister2 provides a hosting environment for high performance data analytics by offering capabilities to connect applications to data platforms such as Kubernetes and Mesos, along with data processing capabilities, including batch and streaming analytics. Twister2 models a complex data application as a dataflow graph. One can configure the different parts of the dataflow graph at different granularities, making it suitable to execute end to end applications in a single dataflow. For example, part of a dataflow graph can be executing a SPIDAL algorithm written according to MPI specification, and another part can be doing streaming analytics.

In previous work, many [40]–[42] have shown that current big data systems do not achieve the performance of HPC frameworks. The (generalized) Bulk Synchronous Processing (BSP) computing model offered by MPI specification is well suited for high performance applications, including Machine Learning. This has been demonstrated by SPIDAL algorithms written both according to MPI specification and Harp. Mostly due

to design and software engineering aspects of MPI implementations, it is difficult to integrate MPI environments with Big data environments.

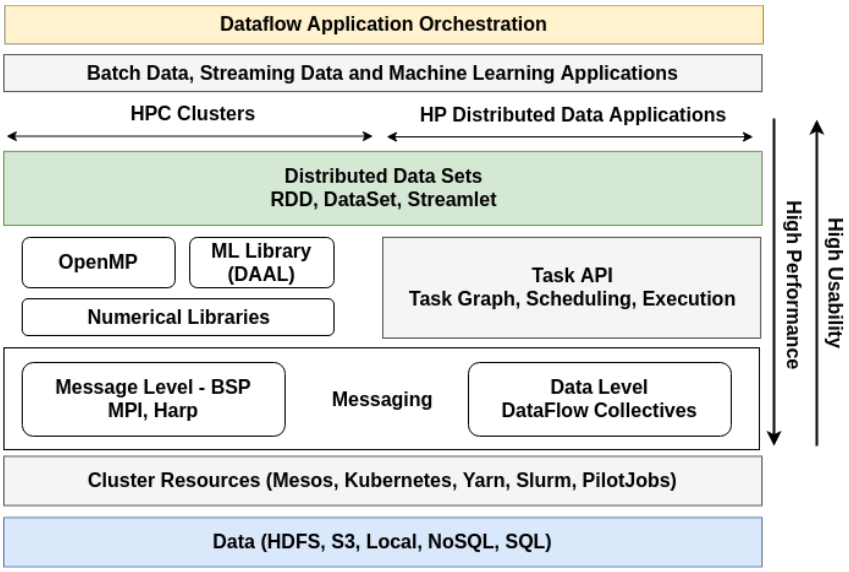

Figure 2-4: Architecture of Twister2. [26], [27]

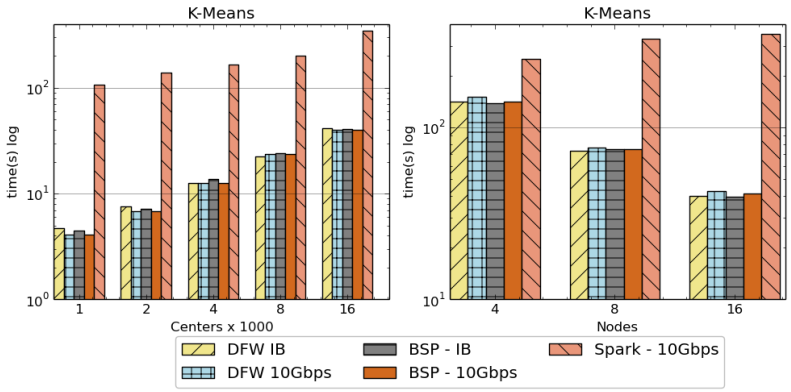

Fig 2-5: Left: K-means job execution time on 16 nodes with varying centers, 2 million points with 320-way parallelism. Right: K-Means with 4,8 and 16 nodes where each node has 20 tasks. 2 million points with 16000 centers used.

The main design goal of Twister2 is to build a component-based architecture for composing data applications that can work both in HPC and cloud environments. Fig. 2-4 shows the complete architecture for integrating machine learning with data analytics using Twister2. At the base, it has abstractions to access different data sources, including file systems, databases, and message brokers. Next, a resource scheduler provides the functionality to allocate cluster resources using different resource managers both in cloud and HPC environments. On top of these layers, it provides communication operations between parallel workers. These communications include both dataflow communications for data pipelines and streaming as well as BSP style

for fine-grained parallel applications. Both task-based and distributed data set based API's are offered to create dataflows that include streaming, batch, and machine learning components. In addition to that, Twister2 will provide fault tolerance at the dataflow node level for both streaming and batch applications.

An optimized dataflow communication has been developed as an independent framework to support both streaming and batch data processing [27] operations. Dataflow communication is used for data processing operations such as shuffle, join unions, and key-based operations and for integrating with other frameworks. This library can work using an MPI implementation or using plain sockets, making it suitable for HPC and clouds. Figure 2-5 shows the performance of K-Means on Spark and Twister2 with varying number of clusters and nodes where it performs considerably well compared to the big data framework. We are taking an incremental approach for developing Twister2 as an open source community driven project. The initial release of Twister2 provides the ability to run both batch and streaming analytics for large datasets. Further, we will enhance the capabilities to provide richer dataflows that include applications written in MPI. The first publicly available Twister2 release was at the end of September 2018 [43] and we expect a near-final version a year later.

3. APPLICATION COMMUNITY ACTIVITIES

3.1. Network Science Activities

3.1.1. Network Generation Systems

Motivation. A number of network science applications involve constructing an ensemble of networks with properties close to a given real network. Such ensembles are useful for sensitivity analysis, significance testing, and scaling studies. There are two broad approaches for generating such random networks. The first involves using a random graph model, e.g., the Erdos-Renyi (ER), Chung Lu (CL), Preferential Attachment model, Stochastic block model (SBM), BTER, etc. There are a large number of such models, and they seek to generate instances, which preserve the degree sequence, and other properties, such as the community structure and clustering. Most of these models are very simple and end up specifying a probability for each pair of nodes to be connected. However, a naïve method of generating instances from such a model would take quadratic time, even if the final graph is sparse. We have developed some of the best scaling algorithms for generating instances from many random graph models in parallel. The second approach involves performing random edge switches, starting at a given graph. This is known to converge to a stationary distribution on the space of graphs with the same degree sequence. However, this is an inherently sequential process, and we have developed parallel algorithms for this problem.

Our Results. In [44], we develop an approach, referred to as the *Degree Grouping* (DG) technique, which enables us to generate very large instances from the ER, CL, SBM and BTER models. The first three models specify a probability for connecting each pair of nodes, and all edges are generated independently. The BTER model involves additional steps to account for clustering constraints. Our approach gives high space and time efficiency. Both, our sequential and parallel algorithms, only require O(D) space, compared to O(n) space required by the previous algorithms, where D is the number of distinct degrees in the given sequence. The running time of our parallel

algorithm is O(m/P + D + P), where m is the expected number of edges, and P is the number of processors. In contrast to earlier algorithms, the associated constants and overheads are significantly smaller for our algorithms. Experimental results show that our algorithms are about 3–4 times faster than the previous algorithms (Fig 3-1). Moreover, our parallel algorithm achieves an almost optimal load balancing using an efficient load balancing technique and scales very well to a large number of processors. Our parallel algorithm can generate a network with 250 billion edges in just 12 seconds using 1024 processors.

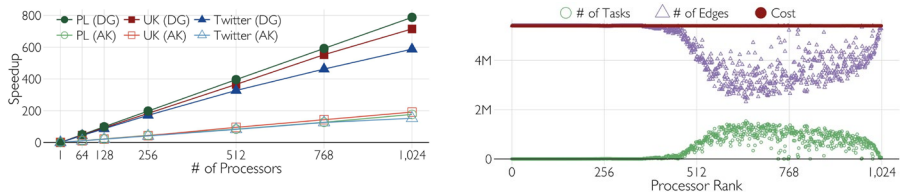

Fig 3-1 Performance of our network generation algorithm (DG) [40], compared with a prior method (denoted by AK): (a) strong scaling, (b) load balancing across processors.

In [45], we develop highly scalable algorithms for edge switching under a variety of constraints. The basic edge switch operation involves picking two random edges (a, b) and (c, d), and replacing them with edges (a, d) and (c, b) respectively, i.e., the end vertices of the selected edges are swapped with each other. This operation is repeated either a given number of times or until a specified criterion is satisfied. It is easy to see that an edge switch operation preserves the degree of each vertex. We present distributed memory parallel algorithms for switching edges in massive graphs with the constraint that the graph remains simple. The dependencies among successive edge switch operations and the requirement of keeping the graph simple lead to significant challenges in designing a parallel algorithm. Dealing with these requires complex synchronization and communication among the processors, which in turn makes it challenging to gain any speedup by parallelization. The performance of the algorithms also depends on the partitioning of the graph. We study several partitioning schemes in conjunction with the algorithms and present their respective trade-offs. A harmonic mean speedup (compared to the sequential algorithm runtime) of 73.25 is achieved on eight different networks with 1024 processors. The algorithms require generating multinomial random variables in parallel, which is also a non-trivial problem. To the best of our knowledge, there is no existing parallel algorithm for the problem, and we present here a novel parallel algorithm for generating multinomial random variables, which achieves a speedup of 925 using 1024 processors.

3.1.2. Subgraph detection and analysis problems

Motivation. A common subroutine in network science involves detecting and counting the number of embeddings of a given (small) subgraph H (e.g., a path or a tree) in a (large) network G. This and other variants are fundamental problems in Network Science and have a wide range of applications in areas such as bioinformatics, social networks, semantic web, transportation and public health. A related problem in network data, arising in anomaly detection using the approach of scan statistics, involves finding connected subgraphs, which maximize different kinds of anomaly score functions. These problems are all computationally challenging to solve exactly

(e.g., the counting problems are #P-hard). Prior methods for counting paths and trees only handled trees of size up to 10 in networks with 2 billion edges.

Our Results. We have developed new parallel algorithms that can handle larger subgraphs. In [34], we develop an improved parallelization of a technique known as color-coding, which has been the primary approach for obtaining rigorous approximation bounds for subgraph counting. Our algorithm, HARPSAHAD+, adapts a prior Hadoop based algorithm to HARP, and yields two orders of magnitude improvement in performance, as a result of its flexibility in task scheduling, data flow control and in-memory cache. We are therefore able to scale to networks with up to billions of edges and obtain a comparable performance when compared to a state-of-the-art MPI/C++ implementation.

For subgraphs of size k, the running time and space complexity of the color-coding technique scale as $O((2e)^k f(n))$ and $O(2^k g(n))$, respectively, where f( ) and g( ) are polynomial in the total number of nodes and edges. Therefore, the color-coding technique has not been able to scale beyond subgraphs of size 12. In [46], we consider a different approach for subgraph detection, based on an algebraic technique called multilinear detection. This technique involves representing subgraphs as multivariate polynomials. The subgraph detection problem is reduced to determining whether the resulting polynomial has a square-free (i.e., multilinear) term. We develop the first MPI based parallel adaptation of this technique, and this gives a significant improvement in both running time and space, which scale as 2^k and k, respectively. This has allowed us to consider subgraphs of size up to 18.

3.2. Biomolecular Simulation Activities

About one-quarter of the computing power provided by XSEDE between 2011 and 2017 was used to perform biomolecular simulations (Figure 3-2). In the biomolecular simulation community *analysis* of the output from simulations is increasingly becoming a bottleneck compared to the generation of the data [47], [48]. In order to address this challenge, **we investigated and developed applications of high-performance data analytics to processing and analyzing biomolecular simulations**, in particular widely used classical molecular dynamics (MD) simulations.

Biomolecular simulations model the dynamics of biomolecules (proteins, nucleic acids, carbohydrates, lipids) in realistic environments such as solvated in water with realistic ion concentrations or a membrane protein embedded in a lipid membrane. They generate as output time series of particle positions (and sometimes also velocities), called trajectories. Data are stored in regular time intervals (time steps) as frames of coordinates. Trajectories are typically analyzed on a frame-by-frame basis. Using statistical mechanics approaches, the positions (and possibly velocities) are used to evaluate the observables of interest [49], [50]. Current simulations typically contain 10^4–10^6 particles (such as atoms in protein and surrounding water) with trajectories containing 10^4–10^7 frames. Thus typical trajectory sizes can span tens of gigabytes to tens of terabytes. Additionally, trajectories are often produced in ensembles of tens to hundreds of trajectories, and often these trajectories are produced in a coupled fashion (for instance, from replica exchange or free energy perturbation simulations). The large amount of data necessitates high-performance data analytics approaches that can use existing high-performance computing resources in order to accelerate the time to solution for the analysis of the computational experiments.

We evaluated a number of data analytics frameworks for their suitability for biomolecular simulations (Sec. 4.4.1) and compared them for typical trajectory analysis algorithms against each other and an MPI-based implementation. The resulting picture is multi-faceted and provides a decision framework for users to choose a suitable data analytics framework. As part of this work, we also developed and implemented production grade parallel versions of algorithms for which we previously only had serial versions (Sec. 4.4.2) and a new library (*PMDA*, "parallel MDAnalysis") that provides a platform for these algorithms. We also identified a barrier to achieving better scaling in parallel trajectory analysis, namely the I/O, and provide proof-of-concept solutions to overcome this problem (Sec. 4.4.3).

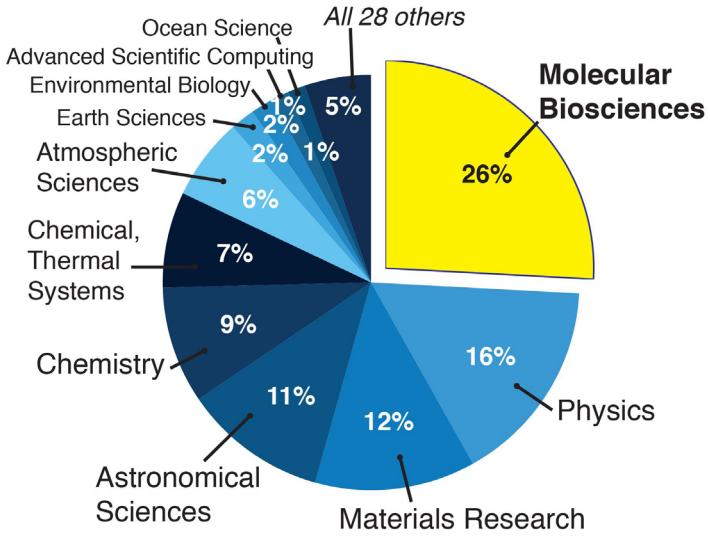

Figure 3-2. XSEDE SUs by research field for the time interval 2011-07-01 to 2017-09-30 [51]

3.3. Image Processing: Polar/Remote sensing

Technology for collecting, transmitting, and storing remote sensing data has rapidly improved over the last decade. In the case of polar science, for example, it is now possible for aircraft equipped with ground-penetrating radar systems to collect large-scale data about a vast area of polar ice in the span of a single flight. But while collecting this data has largely been solved, actually analyzing the data to extract useful data has remained a challenge. Originally, feature tracking in polar radar echograms was a manual process which was occasionally augmented by very simple semi-automated algorithms that usually considered only a few pixels of the image at a time. For 3D imaging, images were truncated to high signal to noise ratio regions with clear interfaces so that the simple automated algorithms could simply track a peak from column to column in the images. Reliance on human annotators and simple algorithms limits the scale and speed at which analysis of remote sensing data can be used.

We have introduced several complementary algorithms that aim to automatically segment polar ice echograms, identifying the regions corresponding to air, ice, and bedrock, and the fine-grained layers between them [52]–[55]. All of these techniques

use machine learning and statistical inference algorithms to robustly handle very noisy data. Our early work [52], [53] investigated this segmentation problem in 2d, along the flight line of the aircraft. Our more recent work [54], [55] has developed new techniques to reconstruct the bedrock layer in 3d, by integrating evidence across many tomographic slices collected along the flight line.

In two of the algorithms, the problem is formulated as an inference problem on a Markov Random Field (MRF), as shown in Fig. 3.3. MRF inference is NP-hard in the general case, so we developed two approximate inference algorithms. The first uses the Viterbi algorithm [56] to solve for individual layers at a time in individual images, which can be done exactly and efficiently. This corresponds to considering a single column at a time of the 3D image MRF shown in Fig. 3.3. However, although the solutions it finds are exact within single layers, the results are not as good on 3D images because the 3D image must be broken into 2D "slices" and the inter-slice dependence cannot be adequately captured by the algorithm. To better handle 3D images, we introduced an approximate inference technique based on Tree Reweighted Message Passing (TRW-S) [54] which can incorporate evidence from multiple slices at once and yields significantly better results than Viterbi. An advantage of the statistical formulation of these models is that additional weak evidence, e.g., data from hand annotators or digital elevation maps, can be incorporated into the inference process. Additional pre-processing steps and additional evidence was added to the cost function used by the Viterbi and TRW-S algorithms, and the model parameters for each were tuned in [57].

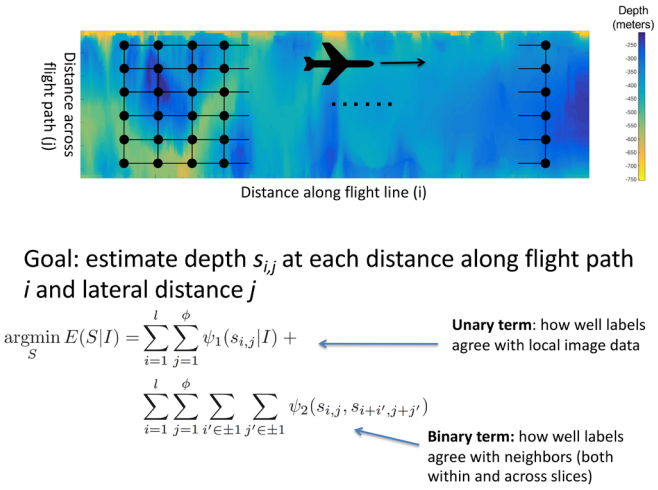

Fig. 3.3: MRF energy minimization function and illustration for 3D ice bed reconstruction problem.

Inspired by the success of deep machine learning across a wide range of problems, we recently [55] introduced a multi-task spatiotemporal neural network (Figure 3.4) that combines 3D Convolutional Networks and Recurrent Neural Networks (RNNs) to address the 3d layer reconstruction problem. This is a novel approach adapted from techniques to analyze video, but there the temporal dimension is treated as equivalent to the row dimension in Fig. 3.4. This algorithm solves for multiple layers simultaneously (rather than one at a time) and improves results significantly over the TRW-S algorithm, even when no extra evidence besides image intensities are available. Depending on the number of iterations required by TRW-S to converge, the neural network approach can also provide a significant speed up.

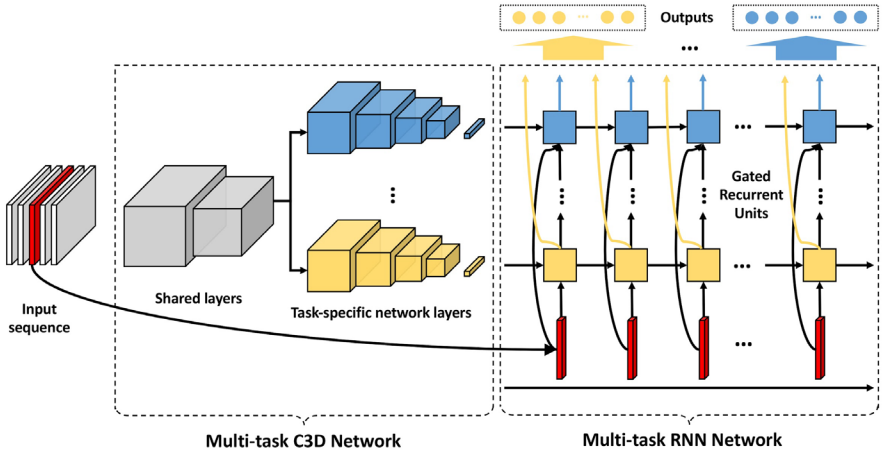

Fig. 3-4: Multi-task spatiotemporal neural network combining 3D Convolutional and Recurrent Neural Networks.

3.4. Scalable Pathology Image Analysis

3.4.1. Introduction to Pathology Image Analysis

The rapid advancement in large-throughput tissue scanning technologies has enabled the production of pathology imaging big data at cellular and subcellular levels with unprecedented rich information. Digital pathology has been used for basic research in a wide scope and becomes an emerging technology promising to support computer-aided diagnosis, which has been recently approved by the FDA [58]. It will enable researchers to better understand the underlying biological mechanisms of pathological evolutions and disease progressions through quantitative pathology analysis and spatial analytics. Recently, 3D digital pathology is made possible through slicing tissues into serial thin sections. The information-lossless 3D tissue space represented by pathology image volumes holds significant potential to enhance biomedical studies and diagnosis through 3D image and spatial analytics. Pathology image analysis can identify massive spatial objects of interest, such as cells and blood vessels, and segment their boundaries. The derived objects and their features are used for complex queries and analytics to support biomedical research or diagnosis.

Major challenges. Digital pathology images are produced at an extremely high resolution. A typical 2D pathology image may contain 100,000 x 100,000 pixels, with a million micro-anatomic objects. A typical 3D tissue volume may generate hundreds of slices, and contain tens of millions of 3D biological objects, and each object could be represented with hundreds to thousands of mesh facets. Such a typical study may involve hundreds of subjects and tens or hundreds of tissue volumes. This unprecedented scale of 2D and 3D data poses significant challenges on data processing and image analysis, leading to tremendous I/O, communication, and computational cost. Besides the scale, 3D spatial objects can have complex structures. For example, 3D blood vessels can have many bifurcations. This poses major challenges to group biological meaningful 2D structure cross-sections for complex 3D structure reconstruction. As a large number of 3D pathology structures often appear in serial 2D image planes, the establishment of 2D cross-section association across serial slides is a significant challenge, especially when structures have complex morphology and

topology. For spatial queries, while minimal bounding boxes (MBBs) have been successfully used in traditional spatial indexing, MBBs are not effective to represent such complex 3D objects in distance-based spatial queries, such as nearest neighbor search. This demands new indexing approaches.

Our goal is to create a framework with new systematic methods to support effective and scalable 2D and 3D pathology image analysis leveraging big data computing platforms and large memory.

3.4.2. Pathology structure segmentation.

We created a robust two-step segmentation framework that can recover contours of an arbitrary number of histologic objects in occlusion.

Segmentation Initialization. For small-scale objects, such as cells, we detect cell seeds with joint information drawn from spatial connectivity, edge map, and shape analysis. For large pathology structures like vessels, we use an adaptive color decomposition method that can dynamically optimize the color decomposition matrix in a way such that the resulting individual stain image channels have the maximum joint entropy. With the derived stain-specific image channel, we create a pathology structure probability map with multi-scale steerable filter processing to indicate the likelihood of a given 2D pixel (or 3D voxel) belonging to a specific pathology structure.

Contour Deformation. We develop a variational level set model that incorporates image intensity, prior shape, and edge information for contour deformation. For cells, we deform contours by minimizing a new energy functional that incorporates a new shape term in a sparse shape prior representation, an adaptive contour occlusion penalty term, and a boundary term encouraging contours to converge to strong edges. A region-based active contour model is developed with a level set formulation based on Mumford-Shah's functional. We create a sparse shape prior library with mode detection by non-parametric clustering over shape manifolds derived from manifold learning. In this way, we can effectively extract representative shape codes from a large number of shape annotations without blindly specifying prior shape number and shape codes in the shape library [59].

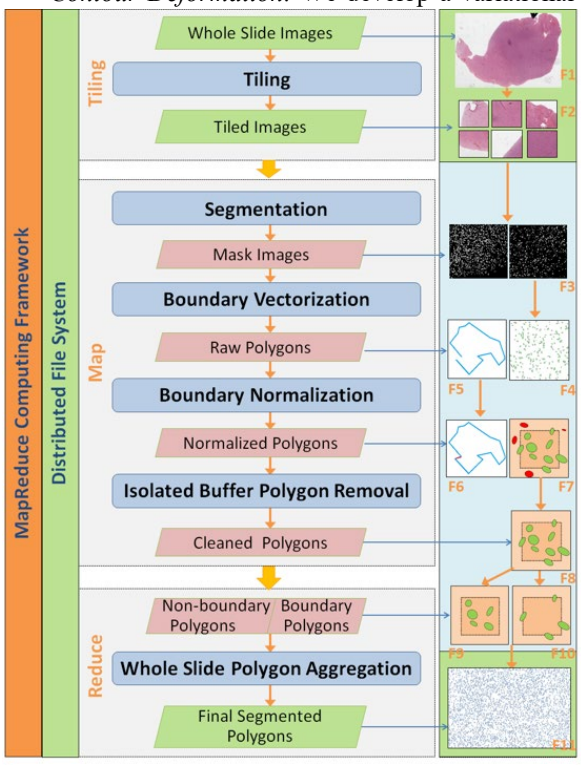

Fig. 3-5. MapReduce based image processing pipeline

3.4.3. Scalable MapReduce based image analysis framework.

To provide scalable pathology image processing, we have developed a highly scalable MapReduce based image analysis framework shown in Fig. 3.5 for whole slide image processing, in which segmentation algorithms can be plugged in. The approach is through partitioning the image space into overlapping tiles for consideration of boundary-crossing object problem, processing titles independently as parallel tasks, and merging and normalizing partial objects from individual tiles. To handle partial objects from overlapping regions, we take a spatial index-based approach for removing redundant polygons that represent the same object. The framework is implemented for both commodity clusters and commercial clouds such as Amazon Web Services (AWS). The performance study showed high scalability on large data sets [60], [61].

3.4.4. 3D pathology image analysis pipeline.

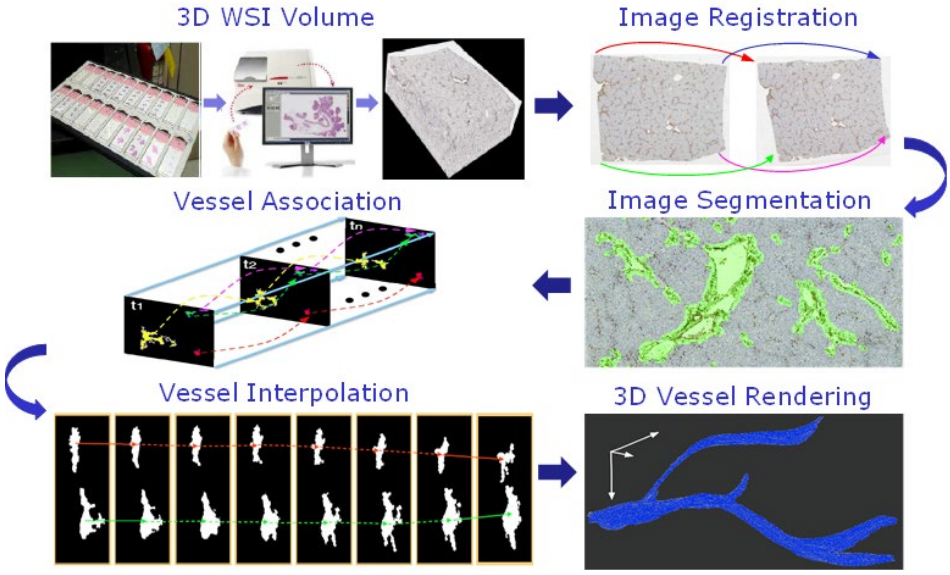

Fig. 3.6. 3D pathology image analysis pipeline

We developed methods on three most principal image analysis components: 3D pathology image registration, pathology structure segmentation, and 3D reconstruction illustrated in Fig. 3.6.

We provided an on-demand registration method within a dynamic multi-resolution transformation mapping and an iterative transformation propagation framework [62], which overcomes the problem of memory limitation on registering extreme large images. This allows us to efficiently scrutinize volumes of interest on-demand in a single 3D space. For segmentation, we develop a scalable segmentation framework for a common set of histopathological structures as discovered above. For 3D reconstruction, we use a novel cross-section association method that solves a geometrical model fitting problem with the statistical multivariate Gaussian distribution for cell reconstruction, and an Integer Programming approach with Markov chain based posterior probability modeling and a Bayesian Maximum A Posteriori (MAP) estimation for 3D vessel reconstruction [63].

3.5. Spatial Big Data Querying Systems

3.5.1. Introduction to Spatial Query Systems

The challenges for spatial big data come not only from the volume but also the complexity, such as the multi-dimensional or dynamic nature of the data. Our research goal on spatial big data management is to address the research challenges for delivering effective, scalable and high performance software systems for managing, querying and mining complex big data at multiple dimensions, including 2D and 3D spatial data, and spatial-temporal data. This is driven by emerging spatial big data problems from geospatial applications, location-based services, and social network applications, and rapid improvement of data acquisition technologies such as high-resolution tissue scanners for pathology imaging. Managing and analyzing such data poses several major challenges, including the explosion of data volume, high complexity of data, and/or temporal dynamics. We have created novel open source software systems shown in Fig. 3.7 for processing 2D and 3D spatial big data on modern big data platforms to support multi-scale applications in various domains, ranging from geospatial data, population data, and microscopy imaging data (biological/pathology objects).

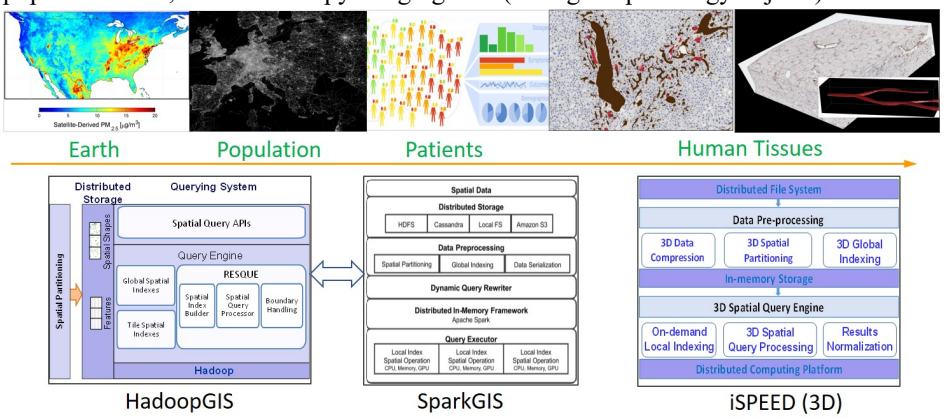

Fig. 3-7. Overview of 3 spatial big data systems described in the text

3.5.2. Hadoop Based Scalable Spatial Queries (Hadoop-GIS)

Hadoop-GIS [64] is a framework with systematic methods to support high performance spatial queries for spatial big data [65], [66]. It provides spatial data processing methods and pipelines with spatial partition level parallelism through simple programming model MapReduce, and take multi-level indexing methods to accelerate spatial data processing. It provided two critical components to enable data parallelism: effective and scalable spatial partitioning in MapReduce (pre-processing), and query normalization methods for partition effect. To provide optimized system performance, we investigate optimization methods for data processing pipelines, such as data skew mitigation, and optimized partitioning methods. The framework is generic, and we have applied the system to support our research on geospatial analytics of social media, GIS-based public health studies, and pathology imaging.

3.5.3. Spark Based Spatial Queries (SparkGIS).

SparkGIS [67], [68] is a spatial data querying system for high throughput and low latency spatial query processing built on Apache Spark. SparkGIS takes advantage of the in-memory processing capabilities of Spark by adapting the core querying methods of Hadoop-GIS to Spark. SparkGIS addresses the problem of memory constraint as querying performance will drop significantly when data does not fit into memory. SparkGIS takes a spatial query rewriting approach to break a big query into small ones that can fit into memory and be pipelined combining compressed binary data storage in memory. SparkGIS supports common spatial queries, including range, spatial join, and k-nearest neighbor search and can be extended to other complex query pipelines. On-demand in-memory indexes allow SparkGIS to prune input data and apply compute-intensive operations on only a subset of relevant spatial objects and consequently achieve higher performance. SparkGIS employs a layered architecture that enables system extensibility for seamless integration of auxiliary distributed computation such as Jaccard Coefficient as a plugin. SparkGIS could significantly boost spatial query performance over base systems.

3.5.4. Efficient In-Memory Based Spatial Queries for Large-Scale 3D Data with Complex Structures (iSPEED).

3D digital pathology imaging is an emerging field that enables novel ways for biomedical research and holds high potential to improve disease diagnosis with an examination of high-resolution 3D image volumes of tissue specimens. Quantitative analyses of 3D pathology images involve exploring spatial relationships among a massive number of biological objects. However, this is challenged by the overwhelming data scale, complex biological structures (such as blood vessels), multiple levels of detail for representations, and high computational complexity for spatial queries and geometric computations. iSPEED [69]–[71] creates effective and scalable in-memory 3D spatial data processing methods with 3D data compression, 3D indexing methods, and partitioning level parallelism for multiple big data platforms including Hadoop and Spark, to support fast discovery of spatial patterns of 3D pathology objects. To achieve low latency, iSPEED stores data in memory with effective progressive compression for each individual 3D object with successive levels of detail, and provides global spatial indexing in memory through effective partitioning. An in-memory 3D spatial query engine will be invoked on-demand to run many instances in parallel. The query parallelization is implemented with, but not limited to, MapReduce and Spark. At run time, iSPEED will dynamically decompress required 3D objects only at the specified LOD, and create necessary spatial indexes in-memory to accelerate query processing, including object-level indexing (inter-objects) and structural indexing (intra-object) on complex structured objects. iSPEED significantly improves 3D spatial query performance compared to traditional disk-based Hadoop based approach.

4. INTEGRATION OF TECHNOLOGY AND APPLICATIONS

4.1. Approaches to a High performance GAIMS

Naively a successful GAIMS must use HPC approaches as it needs to simulate large systems (models) and analyze large data. However, most ABDS and commercial Big Data approaches avoid HPC except for cases like deep learning where HPC is essential and used from the start. However, we intend to pursue better functionality and better performance by combining HPC (including MPI, Pilot Jobs, Slurm) and Big Data (including Hadoop, Spark, Flink, Heron, Docker, Mesos, Kubernetes, and Tensorflow). We can identify different approaches

a) Fix performance issues in Spark, Heron, Hadoop, Flink, etc. This is messy as some features of these big data systems are intrinsically slow while all these systems are "monolithic," and it is difficult to deal with individual components. We and others have explored this for Hadoop, Spark, and Storm.
b) Execute HPBDC from classic big data system with custom communication and task generation environment. This is the approach of Harp for the Hadoop ABDS environment and for RADICAL-Pilot applied to Spark, Hadoop, and Streaming.
c) Provide a native Mesos/Yarn/Kubernetes/HDFS high performance execution environment with the functionalities of Spark, Hadoop, Flink, and Heron. This is the goal of our Twister2 approach.
d) Execute data analytics with MPI in a classic (Slurm, Lustre) HPC environment
e) Improve the classic HPC environments as Pilot Jobs do for task-based execution models.
f) Add modules to existing frameworks like Scikit-Learn or Tensorflow either as new capability or as a higher performance version of existing modules.

We have over the last 8 years explored all these approaches but focus this paper on b) c) and e).

4.2. SPIDAL Parallel Data Analytics Library

The table below lists 34 Core Machine Learning Algorithms implemented in different frameworks with our work in the first two columns. We are packaging these as described in the following section 4.3. SPIDAL also includes community-specific data analytics.

Table 1: 34 Core Machine Learning Algorithms and status in 11 Libraries

	SPIDAL	Harp-DAAL	Intel DAAL	Mahout	Spark	Flink	Turi	Petuum	DMTK	H2O	ANGEL
K-means	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓
MF-SGD	✓	✓		✓				✓			✓
Implicit-ALS	✓	✓	✓	✓	✓	✓	✓				
Neural Network	✓	✓	✓				✓	✓		✓	
PCA	✓	✓	✓	✓	✓					✓	

SVD	✓	✓	✓	✓	✓						
QR	✓	✓	✓	✓							
Covariance	✓	✓	✓								
Linear regression	✓	✓	✓		✓	✓	✓				
Lasso Regression	✓							✓			
logistic Regression	✓			✓	✓		✓	✓			✓
Ridge Regression	✓	✓	✓								
Moments	✓	✓	✓								
Naive Bayes	✓	✓	✓	✓	✓						
Association Rules	✓	✓	✓		✓						
Decision Forest	✓		✓		✓						
K nearest neighbor	✓	✓	✓								
Outlier Detection	✓		✓								
SVM	✓		✓		✓			✓			✓
Random Forest	✓			✓	✓			✓			
Distance Metric Learning	✓							✓			
LDA	✓	✓		✓	✓			✓	✓		✓
GBM (Generalized Linear Modeling)	✓								✓		
Multiverso	✓								✓		
GLM (Generalized Linear Modeling)	✓										
GLRM (Generalized Low Rank Models)	✓										
Stacked Ensembles	✓										
GBDT (Gradient Boost Decision Tree)	✓										✓
Multidimensional Scaling (MDS)	✓										
DA-MDS (Deterministic Annealing-MDS)	✓										
Force-Directed Graph Drawing	✓										
Subgraph Mining	✓										

Irregular DAVS Clustering	✓										
Determ. Annealing Semimetric Cluster	✓										

4.3. Image Processing and Optimization Library and Tutorial

Cheap image sensors, fast networks, and massive storage have made it feasible to collect large-scale collections of images of various types, from photographs on social media sites to remote sensing images taken by radar and satellites, to medical images from x-ray, microscopy, and CT. But while large image datasets can now be easily collected, processing them to automatically extract useful analytics --- identifying objects in the photos, reconstructing properties of the original scenes, segmenting images into different semantic components --- is still a very open problem, attracting a large amount of research across a wide range of fields. All of this activity is, of course, very good for the development of new techniques, but we have observed a large amount of duplication. Scientists and engineers regularly "re-discover" techniques for solving their problems, not realizing that similar problems have already been solved in another community. Not only does this lead to wasted effort, but it means that the customized solutions developed for one particular application may not take advantage of the most advanced theories and implementations from the existing community.

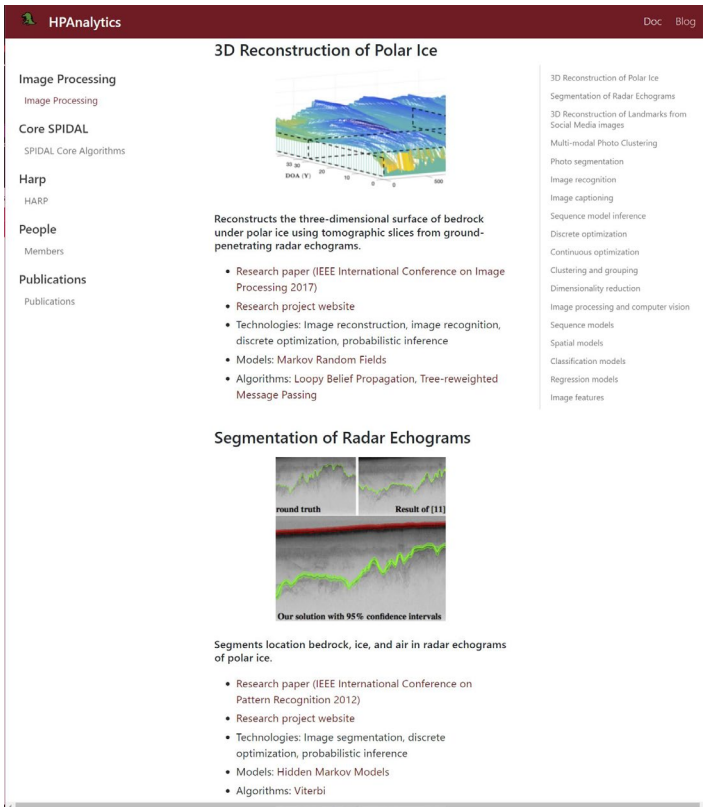

Fig. 4-1: SPIDAL Machine Learning & Image Processing Website

We have identified and developed a library of optimization and image processing techniques that we believe are foundational to a wide range of problems and

applications. The basic philosophy underlying this approach is that many problems in AI-related domains (including machine learning, image understanding, computer vision, etc.) can be posed and solved using one of a relatively small number of general techniques. Instead of re-inventing algorithms that are customized for a particular application, we advocate posing new problems in terms of general and abstract techniques. For example, many problems involving analyzing sequences can be posed as Hidden Markov Models, whether the sequences are sentences, positions of objects across time, audio signals, or protein sequences.

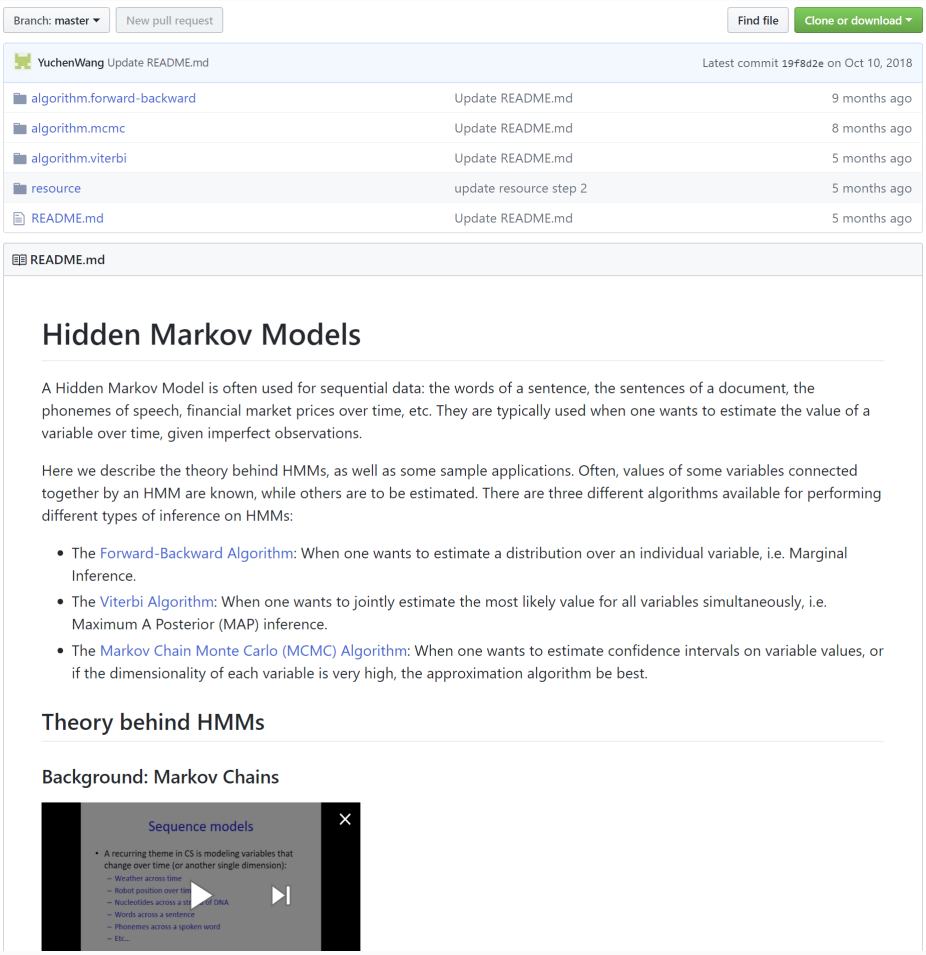

Fig. 4.2: Snapshot of (Hidden Markov) Models section of SPIDAL Machine Learning & Image Processing website showing descriptions and one of the videos.

This view encourages separating applications, models, algorithms, and implementations to maximize generalization and code reuse [72]. An application involves solving a particular domain-specific problem. Many such domain-specific tasks can be posed in terms of a model that precisely defines a mathematical problem to be solved. Different models may make different assumptions, trading off between simplicity and faithfulness to the original problem. An algorithm is a particular strategy for solving for unknown variables in that model. Different algorithms can be used to solve a given model, often with different trade-offs between accuracy and speed. An

implementation is one particular set of source codes for executing an algorithm. Implementations may use heuristics or simplifications to improve practical results or decrease the running time of an algorithm.

This seemingly obvious decoupling of applications, algorithms, models, and implementations can nonetheless be very powerful, because countless applications can be posed in terms of existing models, and new models can be solved with existing algorithms. The models implemented in the library are very general, including sequence model (e.g., Hidden Markov Models), spatial models (Markov Random Fields), classification models (e.g., Nearest Neighbor), feature models (e.g., bag of words, image interest points), etc. The various algorithms can optimize various of these models, each with different trade-offs on the accuracy, running time, and assumptions. For example, implementations of several solvers for Markov Random Field (MRF) models are available, including Markov Chain Monte Carlo (MCMC) which can approximately solve for marginal distributions of MRF variables, Loopy Belief Propagation, which can approximately find Maximum A Posteriori (MAP) solutions to MRFs, and Tree-Reweighted Message Passing (TRW-S), which is also approximate but can offer bounds on accuracy. Other algorithms include the Viterbi algorithm and Forward-Backward for optimization on Hidden Markov Models, Deterministic Simulated Annealing for continuous optimization problems, K-means, Mean-shift, and MultiDimensional Scaling algorithms for clustering and dimensionality reduction, Scale Invariant Feature Transforms for image feature extraction, etc.

Our library tries to remove two major barriers for domain scientists to apply these models and algorithms to their problems. First, we provide implementations of these algorithms so that domain scientists can reuse or modify our implementations instead of starting from scratch. Second, and perhaps even more importantly, we try to help domain scientists pose novel problems in terms of these existing algorithms and models. To do this, we have developed a documentation and tutorial website [73] (see Fig. 4.1) that provides practical information about how the models and algorithms work, which types of problems to apply them on, and which assumptions they make, and so on. We believe this conceptual information may be just as important as documentation about the library routines themselves: the major barrier to applying known models and algorithms is simply not knowing that they exist. The tutorials include slides, videos, links to other references, and source code hosted in GitHub (Fig. 4.2). The tutorial can be used in different ways, including browsing a collection of exemplar applications to see how various routines are combined and applied to particular problems, browsing the selection of Models, browsing the selection of Algorithms, and exploring based on problem type.

4.4. End-to-End Examples of Community Applications with SPIDAL Technologies

4.4.1. Evaluation of task-parallel frameworks for Biomolecular Simulations

Different parallel frameworks for implementing data analysis applications have been proposed by the HPC and Big Data communities. However, it is not clear which ones are suitable for biomolecular simulations. We investigated three task-parallel frameworks, namely Spark [74], Dask [75] and RADICAL-Pilot [37] shown in Figure 4-3, with respect to their ability to support data analytics on HPC resources and compared them to MPI [76].

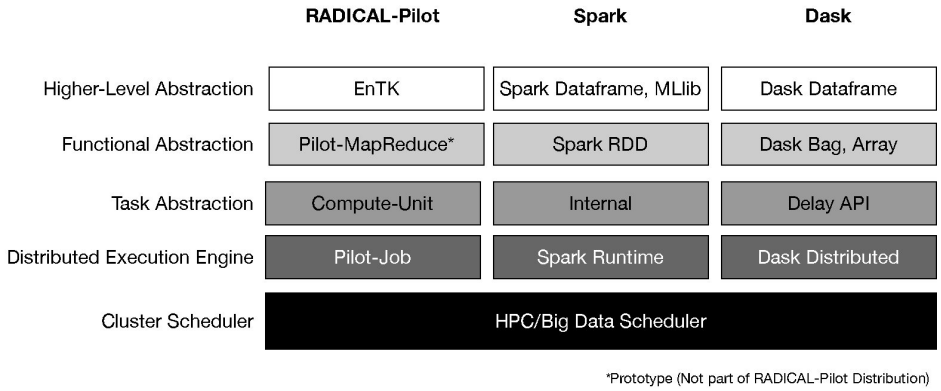

Figure 4-3. The architecture of the Task-parallel data analytics frameworks RADICAL-Pilot, Spark, and Dask, which were evaluated for their suitability for supporting analysis of biomolecular simulations.

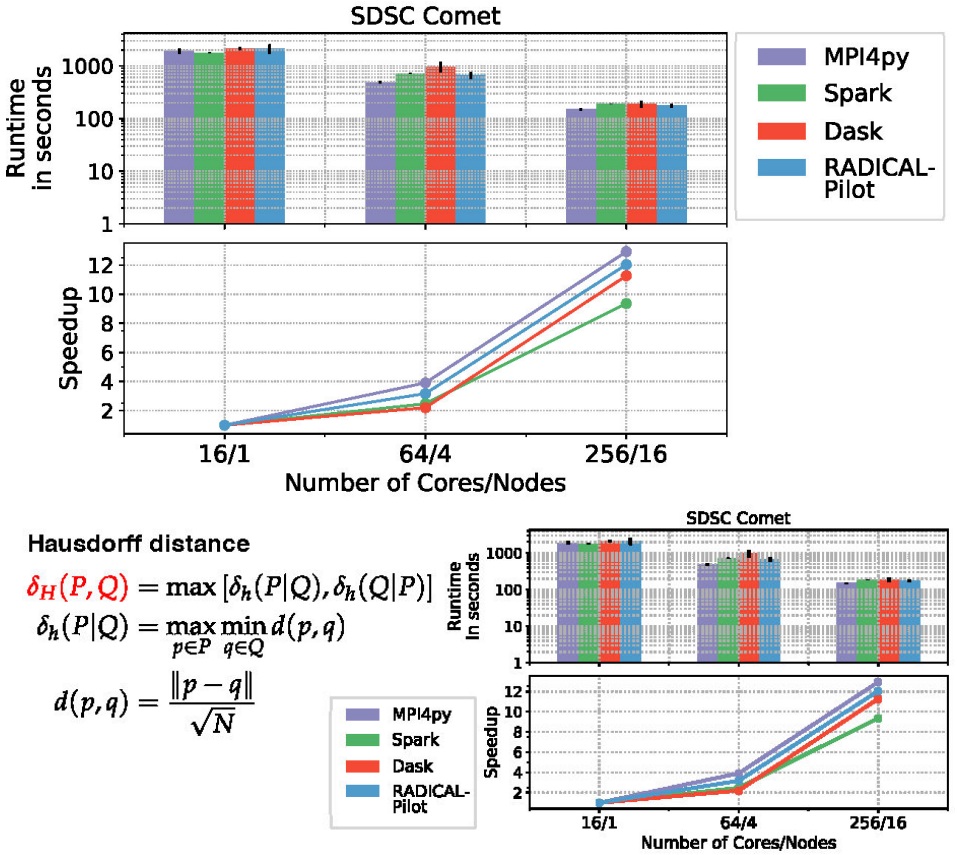

Figure 4-4 Path similarity analysis with the Hausdorff distance $\delta_H(P,Q)$ between trajectories P and Q, consisting of coordinate frames p and q. Strong scaling was studied for 4 different frameworks on SDSC XSEDE facility Comet for N=128 trajectories resulting in 8128 Hausdorff distance calculations.[76]

We studied two algorithms used for the analysis of biomolecular simulations, specifically path similarity analysis (PSA) [77], which quantifies the similarity between

two trajectories, and leaflet identification (LeafletFinder) [78], which is used to identify which lipids belong to one of two layers (leaflets) of a lipid bilayer membrane. Both algorithms are part of the MDAnalysis library [78], [79], which abstracts access to all common trajectory formats inside a Python interface and includes many commonly used analysis algorithms. They exemplify different types of algorithms that are typical in this field. PSA is embarrassingly parallel, whereas LeafletFinder consists of multiple stages that involve distance search for a nearest neighbor graph construction and computation of the connected components of the graph. For the embarrassingly parallel PSA problem, the three task parallel frameworks perform equally well and very similar to an MPI-based solution (implemented with MPI4py), with an example shown in Fig. 4-4.

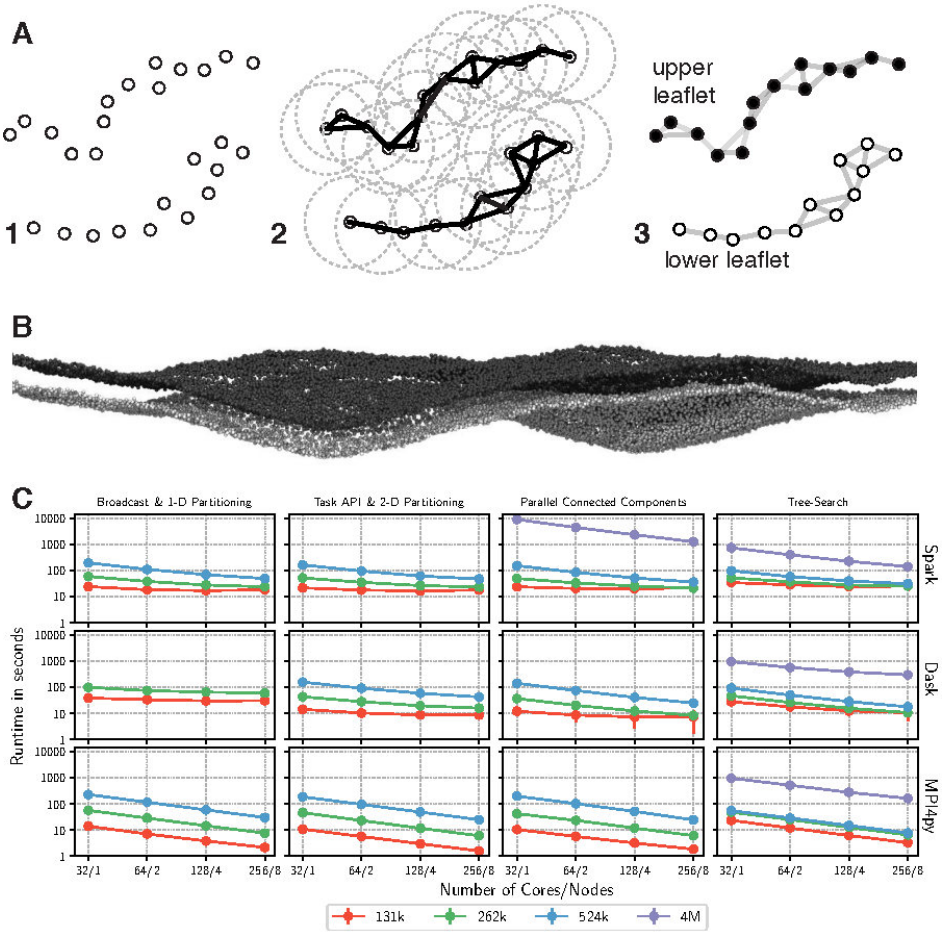

Figure 4-5 Image A from [78] illustrates the LeafletFinder algorithm with B from [78] giving result for curved bilayers. Image C from [76] gives the runtime of 4 parallel implementations on SDSC Comet up to 256 cores (8 full nodes). Spark is top row; Dask the middle and MPI in the bottom. See [76] for RADICAL-Pilot runtime data. Each graph shows different system sizes from 131K to 4M lipids, color-coded as given at the bottom of C. In figs. 4.5A, part 1 shows individual atoms, 2 the graph from their nearest neighbors in a particular cutoff and 3 gives connected components representing two leaflets.

Both the absolute runtime and the strong scaling behavior up to 256 cores are very similar across different machines and data sets of different sizes. For the leaflet identification, four novel different parallel algorithms were implemented [76] and tested (Fig. 4-5). On balance, a map-reduce algorithm with 2D partitioning of the data with a tree-search for the edge discovery combined with partial calculation of connected components performed best across the tested frameworks. The MPI implementation performed best with close to ideal scaling on up to 256 cores (see [76] for the data). Spark was also able to handle large amounts of data well, whereas Dask performed better than Spark for intermediate problem sizes.

Spark, Dask, and RADICAL-Pilot are all suitable for the implementation of these algorithms. For embarrassingly parallel applications, the choice of the framework does not strongly affect the performance. For applications with fine-grained data parallelism, a BigData framework such as Spark and Dask outperforms RADICAL-pilot because of their focus on supporting many short-running tasks. With increasing task communication requirements, using Spark becomes advantageous, and gets close to the performance of MPI. Although the MPI-based implementation always outperformed the task-parallel frameworks, the task-parallel frameworks performed well with good scaling capabilities and in particular, provided a high-level API that makes the implementation of scalable solutions for complex algorithms straightforward. Thus users might nevertheless consider the task-parallel frameworks for their ease of use (especially Dask), provision of high-level abstractions (Spark and Dask), or ease of interoperability with native code (RADICAL-pilot and Dask with Python, Spark with Java).

4.4.2. New algorithms and production-grade implementations for Biomolecular Simulations

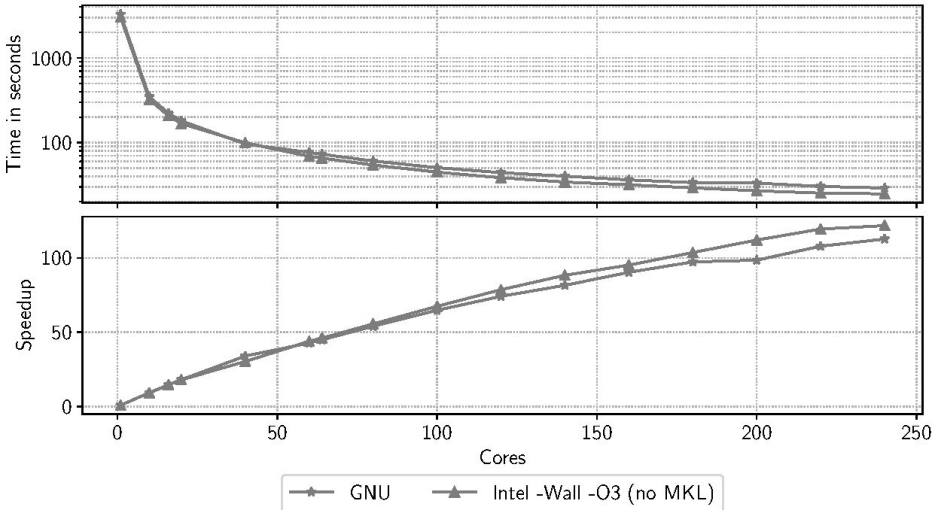

Figure 4-6 Path Similarity Analysis [76] with Hausdorff distance implemented in cpptraj [80]. The algorithm was benchmarked with an ensemble of 128 trajectories for different compilers and ran on up to 240 cores with MPI.

As part of the framework evaluation [76], parallel versions of Path Similarity Analysis with the Hausdorff distance and of the *LeafletFinder* algorithm were developed. The parallel Hausdorff distance calculation was implemented in the popular *cpptraj*

package [80], [81], which is written in C++ and uses OpenMP and MPI parallelism. As shown in Fig. 4-6, the performance is excellent [76]. The Hausdorff distance implementation is built on the "2D RMSD" code in *cpptraj* and only required small additional code changes to make PSA available to the wider cpptraj user community.

The lessons learned during the framework evaluation and our earlier work on parallel trajectory analysis with Dask [82] motivated us to create a new Python-based library for parallel trajectory analysis, named PMDA [83]. PMDA provides a simple API to write parallel trajectory analysis with MDAnalysis by using simple decorators and base classes to leverage existing code in MDAnalysis. The library is still young but important functions such as root mean square distance after optimum structural superposition (RMSD), contact analysis, and radial distribution function are already implemented; the new parallel Hausdorff distance and the tree-based parallel LeafletFinder algorithms from [76] were also implemented in PMDA and are thus are made available to a broad audience.

4.4.3. Identifying and overcoming barriers to progress in Biomolecular Simulations: I/O in parallel trajectory analysis

Our earlier work [82] demonstrated that good performance and scaling could be obtained with a simple MapReduce approach for parallel trajectory analysis (implemented with MPI4py and MDAnalysis) on a single node; however, our approach did not scale beyond a single node (e.g., 72 cores (3 nodes) on SDSC Comet, as shown in Figure 4-7) because a few MPI ranks took much longer to complete than the majority of the ranks (Figure 4-7C).

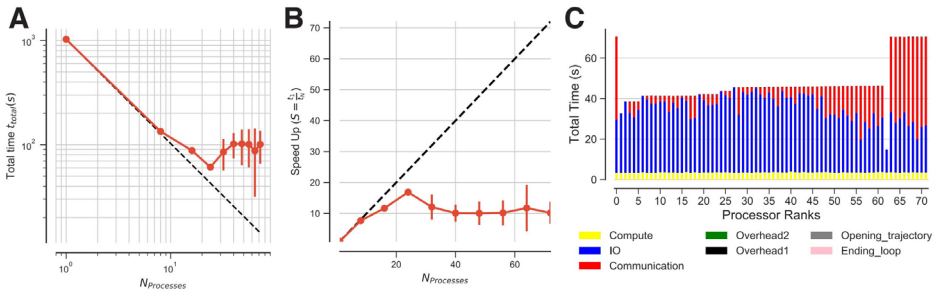

Figure 4-7. Performance [82] of parallel trajectory analysis (RMSD calculation) with serial file access on SDSC Comet. A) Strong Scaling (the total time to completion as a function of the number of processes with one process per core); runs were repeated 5 times, and mean plus standard deviation are shown. Speedup of data in A with ideal scaling indicated by the dashed line in B. Image C shows representative timings of particular MPI ranks up to 72 cores.

The time for reading frames from the trajectory on the shared Lustre file system into memory (I/O time) and the time for communicating results back to rank 0 with MPI.gather() (communication time) fluctuate strongly whereas the time for the actual computation is comparatively small and constant across ranks. This suggested that accessing a trajectory file, which resides on a shared Lustre file system, from multiple ranks might degrade performance.

This effect is especially pronounced when the trajectory analysis is I/O bound and the computational time is comparable to or smaller than the I/O time, as evidenced in Figure 4-8, where better scaling is obtained for higher ratios of the compute to the I/O time.

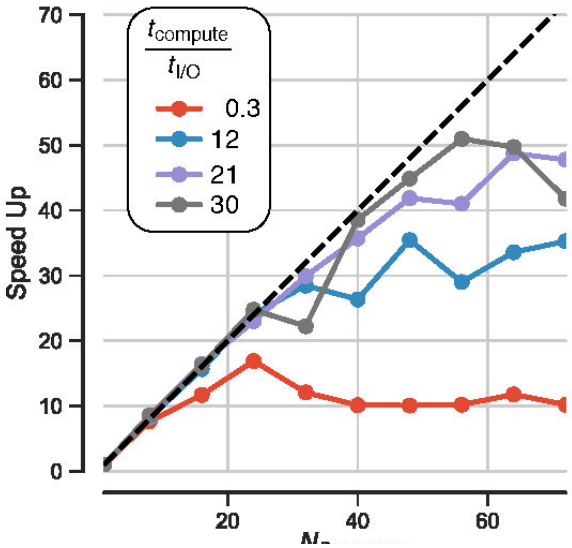

Figure 4.8. Speedup for the RMSD analysis on SDSC Comet for increasing ratio of compute to the I/O time. One MPI Rank was associated with one core up to 3 nodes and cores.

The trajectory in the XTC format (produced by the widely used Gromacs package [84]) is natively read by MDAnalysis [79] using the standard C fopen() call to read the trajectory file. We tested instead parallel MPI-I/O as implemented in the parallel HDF5 library for the HDF5 format. For this proof of concept, we converted the XTC trajectory to HDF5 because even though HDF5 is not widely used in the biomolecular simulation field is one of the few used formats that directly supports integration with MPI-I/O.

Employing MPI-I/O resulted in excellent performance and near-ideal scaling up to 8 nodes (196 cores; see Figure 4-9. The fluctuation in I/O time was minimal, and communication time became negligible (Figure 4-9C). Similar results were also obtained when an XTC trajectory was pre-processed into separate files ("subfiling"), one for each MPI rank, and each MPI rank only read its own file. The latter approach is somewhat cumbersome but can be mitigated by employing a facility in MDAnalysis to read multiple files as a single continuous trajectory. Nevertheless, these initial results indicate that in order to realize the advantages of data analytics on HPC for biomolecular simulations, the trajectory I/O has to be handled carefully.

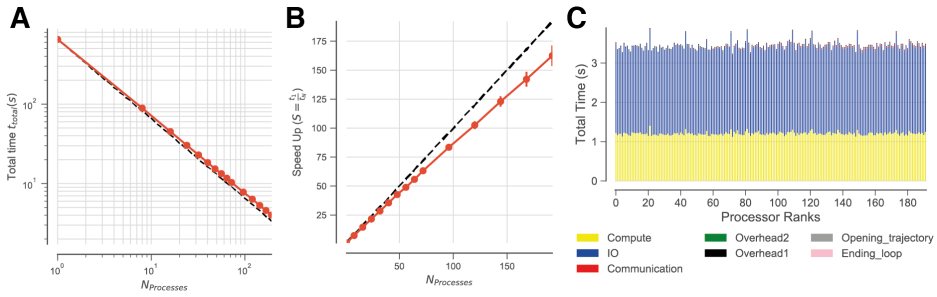

Fig 4-9 Performance of parallel trajectory analysis (RMSD calculation) with parallel HDF5 file access on SDSC Comet. A and B are strong scaling and speedup as in fig. 4-7B. Image C shows representative timings of particular MPI ranks up to 8 nodes (196 cores).

4.4.4. Polar Science as an Application

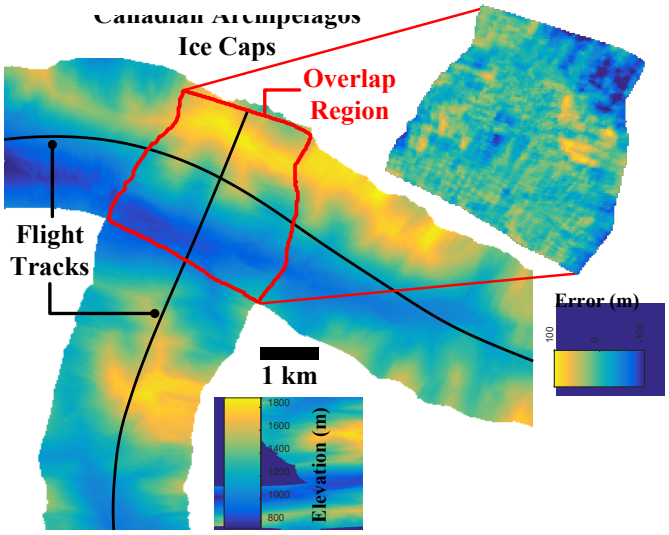

Figure 4.10. Example of two crossing ice bed digital elevation models and the difference error map between them.

As a specific example, the polar remote sensing community is interested in 2D and 3D radar echogram layer and interface tracking of continent-scale ice sheets. Ice sheets are layered depositions which generally flow outward to the margins where ice is removed through calving and melting. Because of the dielectric contrast at the layers and ice interfaces, ground penetrating radars are used to locate the interfaces at the top and bottom of the ice sheet and to detect the internal layer structure. Tracking the top and bottom interfaces of the ice sheet provides essential boundary conditions for realistic ice sheet modeling and is used in ice flux calculations for sea level rise assessments. Tracking the isochronous internal layers allows inference of other useful ice properties (e.g., accumulation rates and basal melting) and can be used to constrain or test the models.

The Viterbi and TRW-S algorithms have been implemented in the open source CReSIS [85] software toolbox [86], which also includes the full processing pipeline from raw data to data product delivery. The algorithms and parameter tuning operations run in parallel and are implemented for several standard computer cluster interfaces (e.g., Torque, Slurm, and Matlab). These algorithms are now in use on a routine basis for tracking the ice bottom [87], [88][57], [89], [90]. The underlying signal processing algorithms have also been improved to produce better image intensity inputs for tracking [91], [92][93].

The data products using the image tracker on 2D data are available at the National Snow, and Ice Data Center [94] for 2016 onwards and the 3D image tracking results are available from the CReSIS data distribution site [95]. An example showing digital elevation models (DEM) of the ice bottom produced from the layer tracking of 3D images is shown in Fig. 4.10 from a mountainous region in the Agassiz Ice Cap. Two DEMs from crossing flight lines are shown. The overlapped portion allows us to test the self-consistency of the DEM generation and the error map based on the difference between the DEMs is shown in the inset.

4.4.5. Network Science Applications

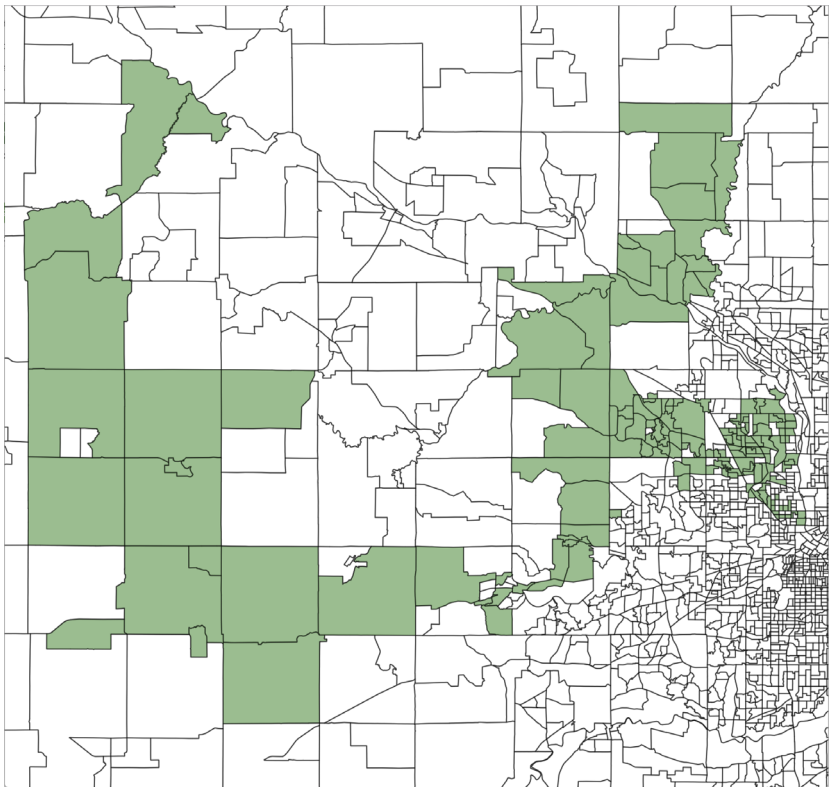

Figure 4-11. A Significant under-immunized cluster found using our network scan statistics approach. The cluster covers the rural blocks west of Minneapolis.

We now discuss an application of our subgraph analysis tools for disease surveillance, which is an important issue in public health. Clusters of under-vaccinated children are emerging in a number of states in the United States due to rising rates of vaccine hesitancy and refusal. As recent outbreaks of diseases such as measles have shown, such clusters can pose a significant public health risk. Such clusters can be used in the epidemiological analysis to determine the risk of outbreaks.

Spatial scan statistics provides a rigorous method for identifying statistically significant clusters. The standard use of this approach (using the SatScan software) involves scanning the entire region using a circular shaped window to find a cluster that optimizes a maximum likelihood objective; this has been extended to other shapes, such as ellipses. However, because of the restriction in shapes, this approach does not find high resolution irregularly shaped regions—if "real" clusters do not have the prescribed shape, SatScan will miss them or only partially catch them.

We use the approach of network scan statistics (for which we have developed more efficient algorithms in [46]) to find such clusters without any restriction on the shape. We run our analysis on the adjacency network G of block groups: the nodes in G are block groups in the region, and two block groups are connected by an edge if they share a boundary. We run the network scan statistic algorithm for the Kulldorff scan

statistic, and find clusters which are more refined, and have a much higher statistical significance, compared to those calculated using SatScan. Figure 4-11 shows an example of such a cluster we find for Minneapolis.

4.4.6. Analysis of Fungal Data from IUB Biology department

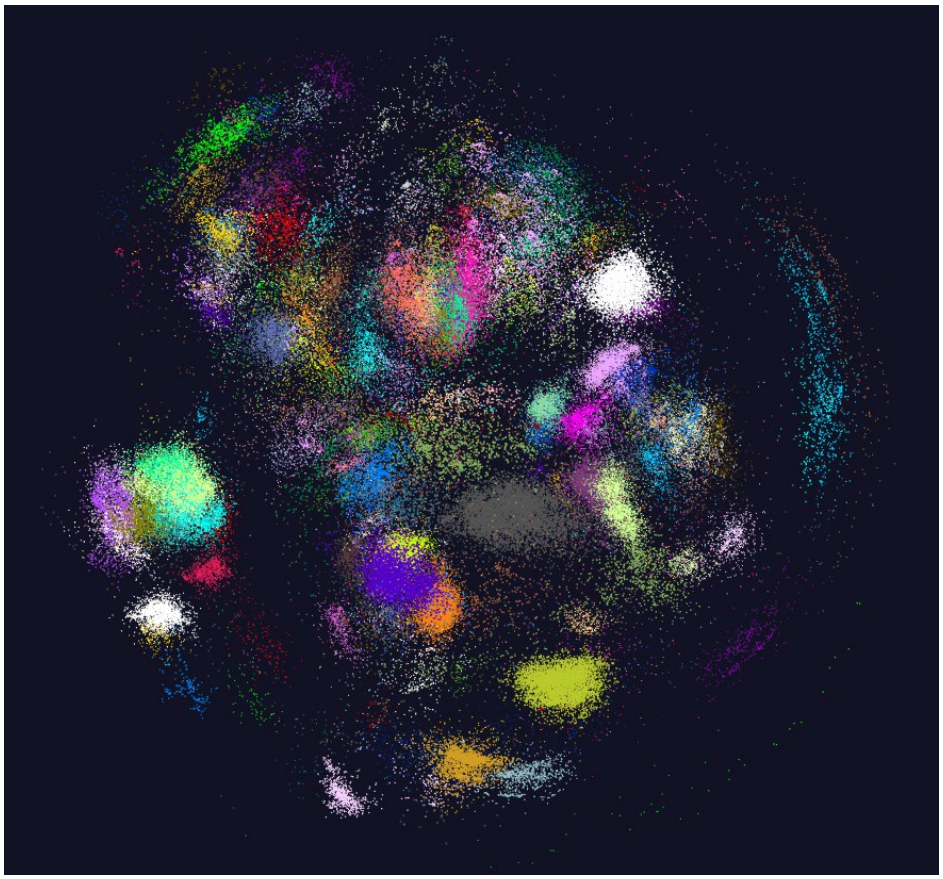

Figure 4-12. Fungi gene sequence data clustered into 65 clusters visualized in 3D space. This is a 2D projection of the 3D dimension reduced visualization.

Identifying structures and patterns in data sets is an important step when analyzing biological data, such as gene sequence data. Correctly identifying and specifying clusters or groups can provide meaningful insights that are not easily captured through the observation of raw data. The difficulty of identifying such clusters increases with the dimensionality of the data set. Therefore, analyzing high dimensional data requires algorithms that can handle high dimensional data for both dimension reduction, as provided through SPIDAL/DA-MDS and clustering as provided through SPIDAL/DA-PWC.

DA-PWC and DA-MDS were successfully used to analyze and identify structures in a set of fungi gene sequence data [Anna Rosling Private Communication]. The total data set, which consisted of roughly 7 million sequences, was pre-processed to identify

roughly 170k unique sequences which occurred more than once within the dataset. DA-PWC was applied iteratively to the 170k dataset, at each iteration, the resulting clusters were visualized by generating 3D data points using DA-MDS and using WebPlotViz [96], [97] to visualize the results.

Visualized results were used to identify and select cluster groups which were to be broken down into sub-clusters in the subsequent iterations. All the visualization steps and managing the workflow between iterations were done through an interactive script which took user inputs after each iteration. Figure 4-12 shows the resulting clusters, visualized in 3D space using WebPlotViz. It is also noteworthy that this method produced superior results when compared to USEARCH (8472 clusters detected) and SWARM (1454 clusters detected) clustering, both of which produced a large number of small insignificant clusters which consisted of just a few (<10) data points.

Additionally, previous research work in collaboration with the IU Biology department and other institutions applied SPIDAL library technologies for various biology data analysis tasks. In [98], [99] SPIDAL technologies were used to analyze rRNA gene sequence structures. In [100] DA-MDS was utilized in order to visualize the Protein Sequence Universe

4.4.7. Pathology and Spatial Analytics for Brain Tumor Microenvironment Studies

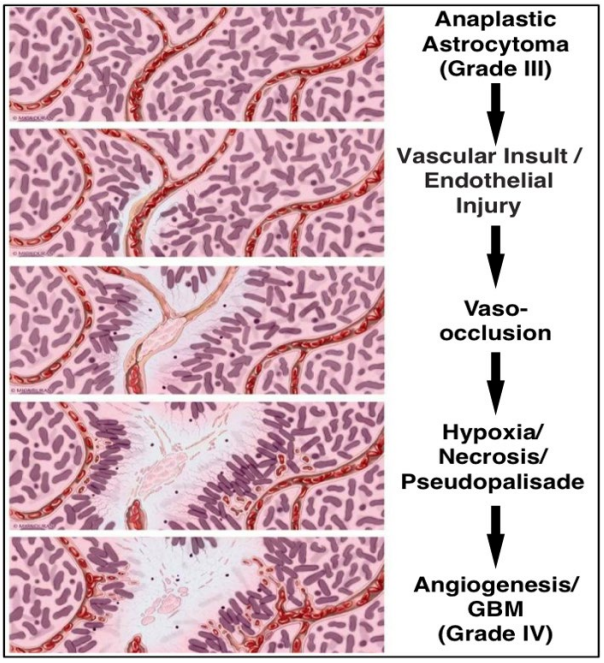

Figure 4-13. Spatial reconstruction during GBM progression.

Glioblastoma (GBM; WHO grade IV) is the most common brain tumor that presents an aggressive clinical progression to death with a more than 90% five-year mortality. The explosive growth properties of GBM contrast sharply with lower grade gliomas (grades II and III). However, the underlying reasons accountable for this abruptly accelerated tumor growth phase are poorly understood. As a result, GBM therapies remain limited in number and efficacy, attributing to a median survival of 12.6 months. Well recognized by domain experts, no current therapy option provides ideal treatment effect due in large part to the limited analysis vehicles to comprehensively and accurately understand GBM histopathology, pathophysiology, and heterogeneity in the context of the tumor microenvironment (TME). Therefore, a thorough and quantitative characterization of GBM TME promoting rapid GBM progression from both histological, and spatially targeted

molecular perspective is the key to create new avenues for novel therapy design and better clinical decision support.

For precise characterizations of genetic, molecular, tissue morphology and clinical biomarkers, the morphological, spatial, and pathophysiological features and interactions of major components in GBM tumor microenvironment require accurate and quantitative analysis support. Structural and morphological features derived from histopathology analysis of the tissue space (in particular, the 3D space) provides rich information of microanatomic objects to understand the tumor microenvironment of GBM. Our work illustrated in fig 4-13, includes reconstructing pathologic objects (vessels and cells) for accurate spatial structure representations, and spatial analytics to characterize 3D spatial patterns of diseases and determine clinical prediction values of their spatial and morphologic features [101].

5. COMPARISON WITH RELATED WORK

5.1. Related work for data analytics and HPC

HPC systems are not able to sustain high I/O rates, which are important for processing the growing datasets. Xuan et al. [102], proposed and developed a new two-level storage system which manages to achieve higher aggregated throughput rates compared to the state-of-the-art. The researchers integrated Tachyon, an in-memory file system with OrangeFS, a parallel file system. The main motivation for using an in-memory file system is that HPC systems often have large memories at each node and they can have a storage capacity comparable to local storage-based HDFS. Moreover, the I/O throughputs of in-memory file systems are much larger than those of local disk. Meanwhile, the parallel file system provides data-fault tolerance and large storage capacity. Therefore this framework combines the best of two worlds. A similar approach to the two-level storage has been proposed in [103]. Their goal is to increase the I/O Performance of Big Data Analytics on HPC Clusters through RDMA-based key-value store. They designed a burst buffer system using RDMA-based Memcached and present three schemes to integrate HDFS with Lustre through this buffer layer. They manage to achieve higher I/O throughput rates by RDMA-based key-value. The reason that they use a key-value store for buffering HDFS I/O on top of the parallel file system is that key-value stores provide flexible APIs to store the HDFS data packets against corresponding block names. Many efforts have been developed to improve the I/O bottleneck of the HPC systems. Researchers have developed the VSFS system [104], which uses a DRAM-based distributed architecture to perform real-time file indexing. Moreover, a versatile index scheme is designed to adapt to the various forms of HPC datasets. The results of our VSFS prototype evaluation show that VSFS is scalable in a typical HPC environment. It achieves significantly better file-indexing and file-search performance than the popular SQL/NoSQL solutions, while it only introduces negligible I/O overhead and has also been tested in real time scenarios with success.

Another major problem that researchers are trying to tackle is the transfer of data from edge sensors. In [105] researchers created a system that uses the MQTT protocol by utilizing the Mosquitto() framework as the data broker of the system and Apache Spark for rapid processing of the data. They found that when the system has many

subscribers, the latency of reading/writing data is increasing, and sometimes the overhead makes it pointless to execute the computations on the compute node.

5.2. Related Work on Communications optimization with HPC and big data

There are fundamental differences between HPC communication as in MPI and data communications required by big data applications. Some of these have been addressed in DataMPI [106], which creates a shuffle engine using MPI primitives. Twister2 tries to identify some of these differences and creates a library called Twister:Net [27], which tries to provide a more generic communication model for data analyses. Exploiting high performance interconnects for data analytics have been studied extensively with HDFS RDMA [107], Spark RDMA [108], Heron Infiniband [109]. In all these instances, existing communication layer of big data tools have been enhanced with high performance interconnects. Recently there have been efforts to implement data operations such as joins using RDMA [110], [111]. Mpignite [112] is an effort to include MPI style communications into the Spark big data framework to improve its collective communication capability.

5.3. Network Science Related Work - Random network generation

A number of random graph models have been developed in an attempt to model different kinds of real-world network data. Efficient parallel generators exist for many of them. For instance, the generator in the Graph 500 benchmark [113] can generate very large instances of the RMAT model [114]. The RMAT model has a simple recursive structure, which makes random generation relatively easy. This is not the case for other stochastic models, such as the Chung Lu (CL), Preferential Attachment (PA) model [115], Stochastic block model (SBM) and block two–level Erdos–Renyi (BTER) models [116]. These models have a more complicated probability structure and a high level of heterogeneity in their degree sequences. Parallel algorithms have been developed for generating instances from these models in both distributed memory systems [44] and MapReduce [116]. The preferential attachment model has a very different structure from the other random graph models——it is an evolutionary model, where edges are added incrementally. Our results in [117] provide a distributed memory parallel algorithm for this model, by exploiting a different but equivalent representation of the PA model. There has been very little work on the second approach of random graph generation through edge switching, and the work of [45] is the first parallel algorithm for this problem.

5.4. Network Science Related Work - Subgraph analysis

There is a vast literature on a variety of subgraph analysis problems, arising out of a number of applications, such as bioinformatics, security, social network analysis, epidemiology, and finance. Here, we focus on the area of subgraph detection and counting, specifically, triangles, paths, and trees, since there has been quite a bit of work on parallel algorithms for these problems.

The triangle counting problem has been studied extensively, because of its connection with clustering coefficient, and applications to social networks, and also due to its local nature. Algorithms have been developed in a variety of distributed computing models, e.g., MapReduce [118], multi-core systems [119], and MPI [120].

These have considered exact counts, as well as deterministic and randomized approximation, e.g., using sampling techniques [121]. In [120], we develop an MPI-based algorithm for exact counting, in the setting where the memory of each machine is large enough to contain the entire network, by dynamic load balancing and tuning task granularity on the fly.

In contrast, the detection and counting of paths and trees are inherently non-local problems. While there are a number of heuristics for these problems, one of the most well-studied methods is based on using the color-coding technique for finding tree-like subgraphs [122], which guarantees a fully polynomial time approximation to the number of embeddings with running time and space, scaling as $O((2e)^k)$ and $O(2^k)$, respectively (with additional terms which are polynomial in the input size). This has been parallelized using MapReduce [123] and OpenMP [124], enabling subgraph counting in graphs with tens of millions of nodes with rigorous guarantees. Slota et al. use threading and techniques for reducing the memory footprint of the color coding dynamic programming tables in order to scale. In [34], [125], we have used the Harp framework to significantly improve the performance of color coding. Finally, in [46], we have developed a new parallel approach for subgraph detection using an algebraic technique.

5.5. Molecular Dynamics Trajectory Data Analysis Related work

Until recently, MD analysis algorithms were executed serially. In the past several years, new frameworks emerged, providing parallel algorithms for analyzing MD trajectories. HiMach [126], by the D. E. Shaw Research group, extends Google's MapReduce for parallel trajectory data analysis. Pteros-2.0 [127] is an open-source library used for modeling and analyzing MD trajectories. Pteros 2.0 parallelizes execution via OpenMP and multithreading, bounding the execution to a single node. MDTraj [128] is a Python package for analyzing MD trajectories, in which parallelization is implemented using the parallel package of IPython as a wrapper. nMoldyn-3 [129] parallelizes the execution through a Master-Worker architecture. The master defines analysis tasks and submits them to a task manager. The task manager distributes tasks to workers.

In contrast, our approach utilizes more general-purpose frameworks for parallelization, such as RADICAL-Pilot, Spark, and Dask. These frameworks provide higher level abstractions, so any integration with other data analysis methods can be fast and easier. Data parallelization can be done on any level of the execution, not only via data-read instructions. In addition, resource acquisition and management are performed transparently. Finally, because these frameworks do not require any compilation before usage, we avoided any hard dependencies that might exist due to the underlying hardware.

5.6. Polar Science Radar Informatics Related work

Most glaciology work on radar data has relied on human annotators [130], which has significantly limited the speed and scale at which polar data can be analyzed. Early automatic annotation techniques for ground-penetrating radar focused on finding boundaries between layers in 2-d radar echograms, typically using basic image processing techniques like region growing [131], [132], edge detection [133], level sets [134], or active contours [135]. Other papers have proposed techniques based on graphical model inference, which can be less sensitive to noise by explicitly modeling

uncertainty and combining weak sources of evidence [52]–[54], [136]. These techniques can also incorporate additional external sources of evidence, such as sparse human annotations or information from digital elevation maps. More recently, deep machine learning has produced significantly better results due to deep neural networks' ability to learn optimal low-level image features from large-scale training data [55], [137],

5.7. Pathology Image Analysis Related work

Radiology image registration is a matured research field with methods broadly categorized into point-, landmark-, surface-, rigid model- Deformable model-, statistical moment-, and image correlation-based registration [138]–[141]. Also, the multi-resolution and multi-scale approaches are developed [142]–[145]. However, these methods can not be directly applied to pathology image registration. Pathology image registration presents its unique challenges due to the overwhelmingly large data scale, a massive number of objects, and large tissue variations. Most prevalent pathology image processing approaches are bounded by 2D phenotypic structure analysis, with limited capacity for processing 3D pathologic objects in imaging volumes. Although some methods have been developed to process 3D pathology imaging data, they only focus on one processing step with no complete 3D analysis pipeline presented [146]–[156]. In spite of the availability of some commercial software for 3D pathology imaging analysis [157]–[160], they are still in immature stages.

5.8. Spatial Data Management and Queries Related work

Spatial Database Management Systems (SDBMSs) have been developed for managing and querying 3D spatial data in industrial applications, such as OracleSpatial [161], MapInfo Discover 3D [162], and ESRI 3D GIS [163]. However, they have limited scalability as they are built on top of ORDBMS, and come with limited 3D geometry support. Data loading is also a major bottleneck for SDBMS based solutions, especially for large-scale datasets. Recently, several systems have been proposed to support large-scale spatial queries with distributed computing resources using MapReduce [164]–[168]. These systems come with their limitations, for example, Spark based spatial querying systems will suffer the out-of-memory problem, and none of them support 3D data types.

6. CONCLUSION AND FUTURES

When this project started the importance of high-performance big data analytics was still not recognized, but nowadays it is clearly a very "hot" area with substantial industry, government, and academic activities. The continued BDEC [169] initiative [170]–[173] is an illustration of this with the first meeting on their "digital continuum platform" held at Indiana University [174], [175]. In the last stages of our project, we will be focussing on taking our work and making it easier for users to take advantage of this. We have started this with the core machine learning and image processing community with the new resource [73] that gives access to software, tutorials, and documentation with a discussion of examples, models, and algorithms. We intend to

follow this approach with the other community products and recently delivered a major tutorial on Twister2 [176]. We are also studying new ideas such as the "Function-as-a-Service" [177] and Tensorflow [178] to make our software available with Python front ends and backend HPC cloud deployments. We need to add standard software engineering practices in the testing arena.

We are also addressing sustainability with different ideas: 1) an NSF IUCRC in high-performance big data computing; 2) the NSF I-Corps program and 3) The Apache Foundation. We are exploring either adding our software to existing Apache projects or setting up a new Apache High Performance Computing project.

We will combine this packaging of our existing work with outreach to new users and new communities in the use of existing building blocks and adding their new building blocks.

We have recently studied [14], [15], [179] several important distinctly different links between machine learning (ML) and HPC. We defined two broad categories: HPCforML and MLforHPC:

- HPCforML: Using HPC to execute and enhance ML performance, or using HPC simulations to train ML algorithms (theory guided machine learning), which are then used to understand experimental data or simulations.
- MLforHPC: Using ML to enhance HPC applications and systems

This categorization is related to Jeff Dean's "Machine Learning for Systems and Systems for Machine Learning" [16] and Satoshi Matsuoka's convergence of AI and HPC [17]. HPCforML includes a major focus of our current project HPCrunsML or using HPC to execute ML with high performance. MLforHPC has 4 subcategories, including MLafterHPC, which includes our current study of data analysis for biomolecular simulations introduced in section 3.2. We have separately started to consider other subcategories MLaroundHPC where one learns results of large scale computations [180] and achieves much higher performance by using learned surrogates and MLAutotuning where machine learning optimizes system configurations [181]. These MLforHPC capabilities are relevant for many of our SPIDAL communities and can be supported by our MIDAS software. This is an exciting vision for the next steps in our project.

Acknowledgments

This work was partially supported by NSF CIF21 DIBBS 1443054; the Indiana University Precision Health initiative and Intel through the Parallel Computing Center at Indiana University. We extend our gratitude to the FutureSystems team for their support with the infrastructure.

7. CITATIONS

[1] Digital Science Center, "SPIDAL Home Page: CIF21 DIBBs: Middleware and High Performance Analytics Libraries for Scalable Data Science - Scalable Parallel Interoperable Data Analytics Library," 2015. [Online]. Available: http://spidal.org/index.html

[2] G. Fox, D. Crandall, J. Qiu, G. Von Laszewski, S. Jha, J. Paden, O. Beckstein, T. Cheatham, M. Marathe, and F. Wang, "Datanet: CIF21 DIBBs: Middleware and High Performance Analytics Libraries for Scalable Data Science NSF14-43054 Progress Report. A 21 month Project Report," Sep. 2016 [Online]. Available: http://dsc.soic.indiana.edu/publications/SPIDAL-DIBBSreport_July2016.pdf

[3] G. Fox, D. Crandall, J. Qiu, G. Von Laszewski, S. Jha, J. Paden, O. Beckstein, T. Cheatham, M. Marathe, and F. Wang, "NSF 1443054: CIF21 DIBBs: Middleware and High Performance Analytics

Libraries for Scalable Data Science, Poster," presented at the DIBBs18 NSF Workshop, Data Infrastructure Building Blocks (DIBBs) https://dibbs18.ucsd.edu/, Washington DC, 2018 [Online]. Available: http://dsc.soic.indiana.edu/presentations/SPIDALPosterDibbsNSF1443054_0622.pdf
[4] NIST Big Data Public Working Group: Use Cases and Requirements Subgroup, *NIST Big Data Interoperability Framework: Volume 3, Use Cases and General Requirements (NIST Special Publication 1500-3)*, vol. 3. NIST, 2016 [Online]. Available: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-3.pdf
[5] "Big Data and Extreme-scale Computing." [Online]. Available: https://www.exascale.org/bdec/. [Accessed: 19-Sep-2018]
[6] K. Asanovic, 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. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.html
[7] Rob F. Van der Wijngaart, Srinivas Sridharan, and Victor W. Lee, "Extending the BT NAS parallel benchmark to exascale computing," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 2012, pp. 1–9.
[8] NASA Advanced Supercomputing Division, "NAS Parallel Benchmarks," 1991. [Online]. Available: https://www.nas.nasa.gov/publications/npb.html
[9] 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*. National Academies Press, 2013 [Online]. Available: http://www.nap.edu/catalog.php?record_id=18374
[10] Geoffrey C. FOX, Shantenu JHA, Judy QIU, Saliya EKANAYAKE, and Andre LUCKOW, "Towards a Comprehensive Set of Big Data Benchmarks," in *Big Data and High Performance Computing*, Lucio Grandinetti and Gerhard Joubert, Eds. IOS, 2015 [Online]. Available: http://grids.ucs.indiana.edu/ptliupages/publications/OgreFacetsv9.pdf
[11] Geoffrey Fox, Judy Qiu, Shantenu Jha, Saliya Ekanayake, and Supun Kamburugamuve, "Big Data, Simulations and HPC Convergence," in *Springer Lecture Notes in Computer Science LNCS 10044*, New Delhi, India, 2016 [Online]. Available: http://dsc.soic.indiana.edu/publications/HPCBigDataConvergence.pdf
[12] CISCO, "Cisco Global Cloud Index: Forecast and Methodology, 2016–2021 White Paper." [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html. [Accessed: 05-Sep-2018]
[13] Cisco Internet Business Solutions Group (IBSG) (Dave Evans), "The Internet of Things: How the Next Evolution of the Internet Is Changing Everything," 2011. [Online]. Available: http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf. [Accessed: 01-Jun-2017]
[14] Geoffrey Fox, James A. Glazier, JCS Kadupitiya, Vikram Jadhao, Minje Kim, Judy Qiu, James P. Sluka, Endre Somogyi, Madhav Marathe, Abhijin Adiga, Jiangzhuo Chen, Oliver Beckstein, and Shantenu Jha, "Learning Everywhere: Pervasive Machine Learning for Effective High-Performance Computation" [Online]. Available: https://arxiv.org/abs/1902.10810, http://dsc.soic.indiana.edu/publications/Learning_Everywhere_Summary.pdf
[15] Geoffrey Fox, James A. Glazier, JCS Kadupitiya, Vikram Jadhao, Minje Kim, Judy Qiu, James P. Sluka, Endre Somogyi, Madhav Marathe, Abhijin Adiga, Jiangzhuo Chen, Oliver Beckstein, and Shantenu Jha, "Learning Everywhere: Pervasive Machine Learning for Effective High-Performance Computation: Application Background," Feb. 2019 [Online]. Available: http://dsc.soic.indiana.edu/publications/Learning_Everywhere.pdf
[16] Jeff Dean, "Machine Learning for Systems and Systems for Machine Learning," in *Presentation at 2017 Conference on Neural Information Processing Systems*, Long Beach, CA [Online]. Available: http://learningsys.org/nips17/assets/slides/dean-nips17.pdf
[17] S. Matsuoka, "Post-K: A Game Changing Supercomputer for Convergence of HPC and Big Data / AI," 13-Feb-2019 [Online]. Available: https://drive.google.com/file/d/1t_F_shSU-48uDh4FKHhpZQXWIhk-BgJX/view?usp=sharing
[18] "HPC-ABDS Kaleidoscope of over 350 Apache Big Data Stack and HPC Technologies." [Online]. Available: http://hpc-abds.org/kaleidoscope/. [Accessed: 01-Dec-2018]
[19] S. Kamburugamuve, "Ph. D. Qualifying Exam: Survey of Apache Big Data Stack," Dec. 2013 [Online]. Available: http://dsc.soic.indiana.edu/publications/survey_apache_big_data_stack.pdf
[20] Geoffrey Fox, Judy Qiu, Shantenu Jha, Supun Kamburugamuve, and Andre Luckow, "HPC-ABDS High Performance Computing Enhanced Apache Big Data Stack," in *Invited talk at 2nd International Workshop on Scalable Computing For Real-Time Big Data Applications (SCRAMBL'15) at CCGrid2015, the 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*,

2015 [Online]. Available: http://dsc.soic.indiana.edu/publications/HPC-ABDSDescribedv2.pdf
[21] Bingjing Zhang, Bo Peng, and Judy Qiu, "High Performance LDA through Collective Model Communication Optimization," in *International Conference on Computational Science*, 2016, vol. 80, pp. 86–97 [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050916306512
[22] B. Peng, B. Zhang, L. Chen, M. Avram, R. Henschel, C. Stewart, S. Zhu, E. Mccallum, L. Smith, T. Zahniser, J. Omer, and J. Qiu, "HarpLDA+: Optimizing Latent Dirichlet Allocation for Parallel Efficiency," in *10th IEEE International Conference on Cloud Computing (IEEE Cloud 2017), June 25-30, 2017 (IEEE Big Data 2017).*, 2017 [Online]. Available: http://dsc.soic.indiana.edu/publications/HarpLDA%2B%20Optimizing%20Latent%20Dirichlet%20Allocation%20for%20Parallel%20Efficiency.pdf
[23] Bingjing Zhang, Yang Ruan, and Judy Qiu, "Harp: Collective Communication on Hadoop," in *IEEE International Conference on Cloud Engineering (IC2E) conference*, 2014 [Online]. Available: http://grids.ucs.indiana.edu/ptliupages/publications/HarpQiuZhang.pdf
[24] Supun Kamburugamuve, Saliya Ekanayake, Milinda Pathirage, and Geoffrey Fox, "Towards High Performance Processing of Streaming Data in Large Data Centers," in *HPBDC 2016 IEEE International Workshop on High-Performance Big Data Computing in conjunction with The 30th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2016)*, 2016 [Online]. Available: http://dsc.soic.indiana.edu/publications/high_performance_processing_stream.pdf
[25] Saliya Ekanayake, Supun Kamburugamuve, and Geoffrey Fox, "SPIDAL: High Performance Data Analytics with Java and MPI on Large Multicore HPC Clusters," in *24th High Performance Computing Symposium (HPC 2016), , 2016, as part of the SCS Spring Simulation Multi-Conference (SpringSim'16)*, 2016 [Online]. Available: http://dsc.soic.indiana.edu/publications/hpc2016-spidal-high-performance-submit-18-public.pdf
[26] S. Kamburugamuve, K. Govindarajan, P. Wickramasinghe, V. Abeykoon, and G. Fox, "Twister2: Design of a Big Data Toolkit," in *EXAMPI 2017 workshop at SC17 conference*, Denver CO 2017 [Online]. Available: http://dsc.soic.indiana.edu/publications/twister2_design_big_data_toolkit.pdf
[27] Supun Kamburugamuve, Pulasthi Wickramasinghe, Kannan Govindarajan, Ahmet Uyar, Gurhan Gunduz, Vibhatha Abeykoon, Geoffrey Fox, "Twister:Net - Communication Library for Big Data Processing in HPC and Cloud Environments," in *Proceedings of Cloud 2018 Conference*, San Francisco [Online]. Available: http://dsc.soic.indiana.edu/publications/twister_net.pdf
[28] B. Zhang, B. Peng, L. Chen, E. Li, Y. Zhou, and J. Qiu, "Introduction to Harp: when Big Data Meets HPC," Indiana University, Jun. 2017 [Online]. Available: http://dsc.soic.indiana.edu/publications/Harp-Report.pdf
[29] "Intel® Parallel Computing Center at Indiana University." [Online]. Available: https://software.intel.com/en-us/articles/intel-parallel-computing-center-at-indiana-university. [Accessed: 30-Sep-2018]
[30] "Intel® Parallel Computing Center at Indiana University led by Judy Qiu." [Online]. Available: http://ipcc.soic.iu.edu/. [Accessed: 30-Sep-2018]
[31] "Intel Parallel Universe Issue 32 page 31: Judy Qiu, Harp-DAAL for High-Performance Big Data Computing." [Online]. Available: https://software.intel.com/sites/default/files/parallel-universe-issue-32.pdf, http://dsc.soic.indiana.edu/publications/Intel-Magazine-HarpDAAL 10.pdf. [Accessed: 30-Sep-2018]
[32] "Intel® Data Analytics Acceleration Library (Intel® DAAL Code Repository." [Online]. Available: https://github.com/intel/daal. [Accessed: 30-Sep-2018]
[33] L. Chen, B. Peng, B. Zhang, T. Liu, Y. Zou, L. Jiang, R. Henschel, C. Stewart, Z. Zhang, E. Mccallum, Z. Tom, O. Jon, and J. Qiu, "Benchmarking Harp-DAAL: High Performance Hadoop on KNL Clusters," in *IEEE Cloud 2017 Conference*, Honolulu, Hawaii [Online]. Available: http://dsc.soic.indiana.edu/publications/2017CLOUDResearchTrack_12261.pdf
[34] Z. Zhao, L. Chen, M. Avram, M. Li, G. Wang, A. Butt, M. Khan, M. Marathe, J. Qiu, and A. Vullikanti, "Finding and Counting Tree-Like Subgraphs Using MapReduce," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 4, no. 3, pp. 217–230, Jul. 2018 [Online]. Available: http://dx.doi.org/10.1109/TMSCS.2017.2768426
[35] "Tutorial on Harp-DAAL: Welcome to HPCDC Tutorial at SC 2017 A high Performance Machine Learning Framework for HPC-Cloud," in *Intel® HPC Developer Conference (HPCDC) 2017 at SC2017*, Sheraton Denver Downtown Hotel, Denver, Colorado, November 11-12, 2017 [Online]. Available: https://dexterrules.github.io/SC-Demo-17/SC-Demo.html#/
[36] A. Luckow, M. Santcroos, A. Merzky, O. Weidner, P. Mantha, and S. Jha, ""P*: A model of pilot-abstractions," in *IEEE 8th International Conference on e-Science*, 2012 [Online]. Available: http://dx.doi.org/10.1109/eScience.2012.6404423
[37] A. Merzky, M. Turilli, M. Maldonado, M. Santcroos, and S. Jha, "Using Pilot Systems to Execute Many Task Workloads on Supercomputers," *arXiv [cs.DC]*, 27-Dec-2015 [Online]. Available:

http://arxiv.org/abs/1512.08194
[38] A. Luckow, I. Paraskevakos, G. Chantzialexiou, and S. Jha, “Hadoop on HPC: Integrating Hadoop and Pilot-Based Dynamic Resource Management,” in *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7530058/
[39] G. Chantzialexiou, A. Luckow, and S. Jha, “Pilot-Streaming: A Stream Processing Framework for High-Performance Computing,” in *2018 IEEE 14th International Conference on e-Science (e-Science)*, 2018, pp. 177–188 [Online]. Available: http://dx.doi.org/10.1109/eScience.2018.00033
[40] S. Kamburugamuve, P. Wickramasinghe, S. Ekanayake, and G. C. Fox, “Anatomy of machine learning algorithm implementations in MPI, Spark, and Flink,” *Int. J. High Perform. Comput. Appl.*, vol. 32, no. 1, pp. 61–73, Jan. 2018 [Online]. Available: https://doi.org/10.1177/1094342017712976
[41] J. L. Reyes-Ortiz, L. Oneto, and D. Anguita, “Big Data Analytics in the Cloud: Spark on Hadoop vs MPI/OpenMP on Beowulf,” *Procedia Comput. Sci.*, vol. 53, pp. 121–130, Jan. 2015 [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915017895
[42] A. Gittens, A. Devarakonda, E. Racah, M. Ringenburg, L. Gerhardt, J. Kottalam, J. Liu, K. Maschhoff, S. Canon, J. Chhugani, and Others, “Matrix factorizations at scale: A comparison of scientific data analytics in Spark and C+ MPI using three case studies,” in *Big Data (Big Data), 2016 IEEE International Conference on*, 2016, pp. 204–213 [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7840606/
[43] “Twister2 Release,” *Twister2 high performance data analytics hosting environment that can work in both cloud and HPC environments*. [Online]. Available: https://github.com/DSC-SPIDAL/twister2/releases. [Accessed: 26-Oct-2018]
[44] M. Alam, M. Khan, A. Vullikanti, and M. Marathe, “An Efficient and Scalable Algorithmic Method for Generating Large: Scale Random Graphs,” in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, Salt Lake City, Utah, 2016, pp. 32:1–32:12 [Online]. Available: http://dl.acm.org/citation.cfm?id=3014904.3014947
[45] H. Bhuiyan, M. Khan, J. Chen, and M. Marathe, “Parallel Algorithms for Switching Edges in Heterogeneous Graphs,” *J. Parallel Distrib. Comput.*, vol. 104, pp. 19–35, Jun. 2017 [Online]. Available: http://dx.doi.org/10.1016/j.jpdc.2016.12.005
[46] S. Ekanayake, J. Cadena, U. Wickramasinghe, and A. Vullikanti, “MIDAS: Multilinear Detection at Scale,” in *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2018, pp. 2–11 [Online]. Available: http://dx.doi.org/10.1109/IPDPS.2018.00011
[47] Thomas E. Cheatham III and Daniel R. Roe, “The Impact of Heterogeneous Computing on Workflows for Biomolecular Simulation and Analysis,” *Comput. Sci. Eng.*, vol. 17, no. 2, pp. 30–39, 2015 [Online]. Available: http://scitation.aip.org/content/aip/journal/cise/17/2/10.1109/MCSE.2015.7
[48] A. Shkurti, I. D. Styliari, V. Balasubramanian, I. Bethune, C. Pedebos, S. Jha, and C. A. Laughton, “CoCo-MD: A Simple and Effective Method for the Enhanced Sampling of Conformational Space,” *J. Chem. Theory Comput.*, Jan. 2019 [Online]. Available: http://dx.doi.org/10.1021/acs.jctc.8b00657
[49] D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications, Second Edition*. Academic Press, 2002.
[50] C. Mura and C. E. McAnany, “An introduction to biomolecular simulations and docking,” *Mol. Simul.*, vol. 40, no. 10–11, pp. 732–764, Aug. 2014 [Online]. Available: https://doi.org/10.1080/08927022.2014.935372
[51] “XSEDE User Portal (XUP).” [Online]. Available: https://portal.xsede.org/#/gallery. [Accessed: 23-Sep-2018]
[52] David Crandall, Geoffrey Fox, and John Paden, “Layer-finding in radar echograms using probabilistic graphical models,” in *IAPR International Conference on Pattern Recognition*, 2012 [Online]. Available: http://grids.ucs.indiana.edu/ptliupages/publications/icpr12-iceFINAL.pdf
[53] Stefan Lee, Jerome Mitchell, David J Crandall, and Geoffrey C Fox, *Estimating bedrock and surface layer boundaries and confidence intervals in ice sheet radar imagery using MCMC*. IEEE, 2014.
[54] M. Xu, D. J. Crandall, G. C. Fox, and J. D. Paden, “Automatic Estimation of Ice Bottom Surfaces from Radar Imagery,” in *IEEE International Conference on Image Processing ICIP2017*, Beijing, Chins, 17-20 September 2017 [Online]. Available: http://arxiv.org/abs/1712.07758
[55] M. Xu, C. Fan, J. D. Paden, G. C. Fox, and D. J. Crandall, “Multi-Task Spatiotemporal Neural Networks for Structured Surface Reconstruction,” in *IEEE Winter Conference on Applications of Computer Vision (WACV), 2018*, Harvey’s Casino in Lake Tahoe, NV/CA, 2018 [Online]. Available: http://arxiv.org/abs/1801.03986
[56] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009 [Online]. Available: https://dl.acm.org/citation.cfm?id=1795555
[57] V. Berger, M. Xu, S. Chu, D. Crandall, J. Paden, and G. C. Fox, “AUTOMATED TRACKING OF 2D

AND 3D ICE RADAR IMAGERY USING VITERBI AND TRW-S," in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS), 2018*, Valencia Spain [Online]. Available: http://dsc.soic.indiana.edu/publications/20180109042829_276176_4106.pdf
[58] "FDA allows marketing of first whole slide imaging system for digital pathology." [Online]. Available: https://www.fda.gov/newsevents/newsroom/pressannouncements/ucm552742.htm. [Accessed: 23-Sep-2018]
[59] P. Zhang, F. Wang, G. Teodoro, Y. Liang, D. Brat, and J. Kong, "AUTOMATED LEVEL SET SEGMENTATION OF HISTOPATHOLOGIC CELLS WITH SPARSE SHAPE PRIOR SUPPORT AND DYNAMIC OCCLUSION CONSTRAINT," *Proc. IEEE Int. Symp. Biomed. Imaging*, vol. 2017, pp. 718–722, Apr. 2017 [Online]. Available: http://dx.doi.org/10.1109/ISBI.2017.7950620
[60] H. Vo, J. Kong, D. Teng, Y. Liang, A. Aji, G. Teodoro, and F. Wang, "MaReIA: a cloud MapReduce based high performance whole slide image analysis framework," *Distributed and Parallel Databases*, Jul. 2018 [Online]. Available: https://doi.org/10.1007/s10619-018-7237-1
[61] H. Vo, J. Kong, D. Teng, Y. Liang, A. Aji, G. Teodoro, and F. Wang, "Cloud-Based Whole Slide Image Analysis Using MapReduce," in *Proceedings of the Second International Workshop on Data Management and Analytics for Medicine and Healthcare - Volume 10186*, 2017, pp. 62–77 [Online]. Available: https://doi.org/10.1007/978-3-319-57741-8_5
[62] B. J. Rossetti, F. Wang, P. Zhang, G. Teodoro, D. J. Brat, and J. Kong, "DYNAMIC REGISTRATION FOR GIGAPIXEL SERIAL WHOLE SLIDE IMAGES," *Proc. IEEE Int. Symp. Biomed. Imaging*, vol. 2017, pp. 424–428, Apr. 2017 [Online]. Available: http://dx.doi.org/10.1109/ISBI.2017.7950552
[63] Y. Liang, F. Wang, D. Treanor, D. Magee, G. Teodoro, Y. Zhu, and J. Kong, "A 3D Primary Vessel Reconstruction Framework with Serial Microscopy Images," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 2015, pp. 251–259 [Online]. Available: http://dx.doi.org/10.1007/978-3-319-24574-4_30
[64] "Hadoop-GIS website: High Performance Distributed Spatial Data Warehousing and Query Processing System over Hadoop." [Online]. Available: http://bmidb.cs.stonybrook.edu/hadoopgis/index. [Accessed: 24-Sep-2018]
[65] A. Aji, F. Wang, H. Vo, R. Lee, Q. Liu, X. Zhang, and J. Saltz, "Hadoop-GIS: A High Performance Spatial Data Warehousing System Over MapReduce," *Proceedings VLDB Endowment*, vol. 6, no. 11, pp. 1009–1020, 2013 [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/3814183
[66] F. Wang, R. Lee, Q. Liu, A. Aji, X. Zhang, and J. Saltz, "Hadoop-GIS: A High Performance Query System for Analytical Medical Imaging with MapReduce," *Technical Report, CCI-TR-2011-3, Center for Comprehensive Informatics, Emory University.*, 2011.
[67] "SparkGIS website: High Performance Distributed Spatial Data Warehousing and Query Processing System over Apache Spark." [Online]. Available: http://bmidb.cs.stonybrook.edu/sparkgis/index. [Accessed: 24-Sep-2018]
[68] F. Baig, H. Vo, T. Kurc, J. Saltz, and F. Wang, "SparkGIS: Resource Aware Efficient In-Memory Spatial Query Processing," in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, Redondo Beach, CA, USA, 2017, pp. 28:1–28:10 [Online]. Available: http://doi.acm.org/10.1145/3139958.3140019
[69] "iSPEED website: In-Memory Based Spatial Queries for Large-Scale 3D Data." [Online]. Available: http://bmidb.cs.stonybrook.edu/ispeed/index. [Accessed: 24-Sep-2018]
[70] Y. Liang, H. Vo, J. Kong, and F. Wang, "iSPEED: an Efficient In-Memory Based Spatial Query System for Large-Scale 3D Data with Complex Structures," in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2017, p. 17 [Online]. Available: https://dl.acm.org/citation.cfm?id=3139958.3139961. [Accessed: 24-Sep-2018]
[71] Yanhui Liang, Hoang Vo, Jun Kong, Fusheng Wang, "iSPEED: a Scalable and Distributed In-Memory Based Spatial Query System for Large and Structurally Complex 3D Data. A Demo Paper," in *Proceedings of the 44th International Conference on Very Large Data Bases (VLDB 2018) Volume 11 Issue 12,* Rio de Janeiro, Brazil., 2018, pp. 2078–2081 [Online]. Available: http://www.vldb.org/pvldb/vol11/p2078-vo.pdf
[72] S. J. D. Prince, *Computer Vision: Models, Learning, and Inference*. Cambridge University Press, 2012 [Online]. Available: https://market.android.com/details?id=book-PmrICLzHutgC
[73] "SPIDAL Machine Learning & Image Processing Website." [Online]. Available: https://hpcanalytics.github.io/ml-tutorial/. [Accessed: 30-Sep-2018]
[74] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica, "Spark: Cluster Computing with Working Sets," in *2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '10)*, 2010 [Online]. Available: http://www.cs.berkeley.edu/~franklin/Papers/hotcloud.pdf
[75] M. Rocklin, "Dask: Parallel computation with blocked algorithms and task scheduling," in *Proceedings of the 14th Python in Science Conference*, 2015 [Online]. Available: http://conference.scipy.org/proceedings/scipy2015/pdfs/matthew_rocklin.pdf

[76] I. Paraskevakos, A. Luckow, M. Khoshlessan, G. Chantzialexiou, T. E. Cheatham, O. Beckstein, G. C. Fox, and S. Jha, "Task-parallel Analysis of Molecular Dynamics Trajectories," in *Proceedings of the 47th International Conference on Parallel Processing*, Eugene, OR, USA, 2018, pp. 49:1–49:10 [Online]. Available: http://doi.acm.org/10.1145/3225058.3225128
[77] Sean L Seyler, Avishek Kumar, Michael F Thorpe, and Oliver Beckstein, "Path Similarity Analysis: A Method for Quantifying Macromolecular Pathways," *PLoS Comput. Biol.*, vol. 11, no. 10, p. e1004568, 2015.
[78] Naveen Michaud-Agrawal, Elizabeth J. Denning, Thomas B. Woolf, and Oliver Beckstein, "MDAnalysis: A toolkit for the analysis of molecular dynamics simulations," *J. Comput. Chem.*, vol. 32, no. 10, pp. 2319–2327, 2011 [Online]. Available: http://dx.doi.org/10.1002/jcc.21787
[79] R. J. Gowers, M. Linke, J. Barnoud, T. J. E. Reddy, M. N. Melo, S. L. Seyler, D. L. Dotson, J. Domanski, S. Buchoux, I. M. Kenney, O. Beckstein, "MDAnalysis: A Python package for the rapid analysis of molecular dynamics simulations," in *Proceedings of the 15th Python in Science Conference, Austin, TX, 2016. SciPy.*, Austin, TX, 2016, pp. 102 – 109.
[80] "CPPTRAJ wiki." [Online]. Available: https://github.com/Amber-MD/cpptraj/wiki
[81] D. R. Roe and T. E. Cheatham, "PTRAJ and CPPTRAJ: Software for Processing and Analysis of Molecular Dynamics Trajectory Data," *J. Chem. Theory Comput.*, vol. e-pub, 2013 [Online]. Available: http://dx.doi.org/10.1021/ct400341p
[82] M. Khoshlessan, I. Paraskevakos, S. Jha, O. Beckstein, "Parallel analysis in MDAnalysis using the Dask parallel computing library," in *Proceedings of the 16th Python in Science Conference, SciPy.*, Austin, TX, 2017, pp. 64–72.
[83] "Python-based library for parallel trajectory analysis PMDA." [Online]. Available: https://github.com/hpcanalytics/pmda. [Accessed: 23-Sep-2018]
[84] M. J. Abraham, T. Murtola, R. Schulz, S. Páll, J. C. Smith, B. Hess, and E. Lindahl, "GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers," *SoftwareX*, vol. 1–2, pp. 19–25, Sep. 2015 [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352711015000059
[85] Center for the Remote Sensing of Ice Sheets (CReSIS), "CReSIS Homepage." [Online]. Available: https://www.cresis.ku.edu/
[86] "Open source CReSIS software toolbox." [Online]. Available: https://git.cresis.ku.edu/ct/. [Accessed: 25-Sep-2018]
[87] J. Paden, M. Xu, J. Sprick, S. Athinarapu, D. Crandall, D. O. Burgess, M. J. Sharp, G. C. Fox, and C. Leuschen, "3D Imaging and Automated Ice Bottom Tracking of Canadian Arctic Archipelago Ice Sounding Data," in *American Geophysical Union (AGU) Fall Meeting, 2016*, San Francisco [Online]. Available: http://dsc.soic.indiana.edu/publications/Paden_C13C_0846_3D_Imaging.pdf
[88] J. Paden, T. Stumpf, and M. Al-Ibadi, "Wideband DOA Estimation for Ice Sheet Bed Mapping," in *IEEE International Symposium on Phased Array Systems and Technology (PAST)*, Waltham, MA [Online]. Available: https://www.researchgate.net/publication/317588565_Wideband_DOA_Estimation_for_Ice_Sheet_Bed_Mapping
[89] M. Al-Ibadi, J. Sprick, S. Athinarapu, T. Stumpf, J. Paden, C. Leuschen, F. Rodríguez, M. Xu, D. Crandall, G. Fox, D. Burgess, M. Sharp, L. Copland, and W. Van Wychen, "DEM extraction of the basal topography of the Canadian archipelago ICE caps via 2D automated layer-tracker," in *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2017, pp. 965–968 [Online]. Available: http://dx.doi.org/10.1109/IGARSS.2017.8127114
[90] M. Al-Ibadi, J. Sprick, S. Athinarapu, V. Berger, T. Stumpf, J. Paden, C. Leuschen, F. Rodríguez, M. Xu, D. Crandall, G. Fox, D. Burgess, M. Sharp, L. Copland, and W. Van Wychen, "Crossover analysis and automated layer-tracking assessment of the extracted DEM of the basal topography of the canadian arctic archipelago ice-cap," in *2018 IEEE Radar Conference (RadarConf18)*, 2018, pp. 0862–0867 [Online]. Available: http://dx.doi.org/10.1109/RADAR.2018.8378673
[91] S. Athinarapu, "Model Order Estimation and Array Calibration for Synthetic Aperture Radar Tomography, MS Thesis, U. Kansas, 2018," Masters, Kansas University, Electrical Engineering, 2018.
[92] Theresa M. Stumpf, "A Wideband Direction of Arrival Technique for Multibeam, Wide-Swath Imaging of Ice Sheet Basal Morphology," University of Kansas, 2015.
[93] S. Athinarapu, J. Paden, M. Al-Ibadi, and T. Stumpf, "Model order estimators using optimal and suboptimal methods with numerical tuning," in *2018 IEEE Radar Conference*, Oklahoma City, 2018 [Online]. Available: http://radarconf18.org/pdf/Detailed_AGENDA_from_the_2018_Proceedings.pdf
[94] "National Snow and Ice Data Center." [Online]. Available: https://nsidc.org/icebridge/portal. [Accessed: 26-Sep-2018]
[95] "CReSIS data distribution site." [Online]. Available: https://data.cresis.ku.edu/. [Accessed: 26-Sep-2018]

[96] Digital Science Center, “WebPlotViz: A tool for visualizing large and high-dimensional data.” [Online]. Available: https://spidal-gw.dsc.soic.indiana.edu/. [Accessed: 13-Feb-2018]
[97] Supun Kamburugamuve, Pulasthi Wickramasinghe, Saliya Ekanayake, Chathuri Wimalasena, Milinda Pathirage, and Geoffrey Fox, “TSmap3D: Browser Visualization of High Dimensional Time Series Data,” in *Advances in High Dimensional Big Data (2nd Workshop) in 2016 IEEE International Conference on Big Data*, Washington DC, USA, 2016 [Online]. Available: http://dsc.soic.indiana.edu/publications/tsmap3d.pdf
[98] Geoffrey L. House, Saliya Ekanayake, Yang Ruan, Ursel Schütte, Wittaya Kaonongbua, Geoffrey Fox, Yuzhen Ye, and James D. Bever, “Phylogenetically structured differences in rRNA gene sequence variation among species of arbuscular mycorrhizal fungi and their implications for sequence clustering”, , , June 3 2016,” *Appl. Environ. Microbiol.*, 2016.
[99] Yang Ruan, Geoffrey L. House, Saliya Ekanayake, Ursel Schütte, James D. Bever, Haixu Tang, and Geoffrey Fox, “Integration of Clustering and Multidimensional Scaling to Determine Phylogenetic Trees as Spherical Phylograms Visualized in 3 Dimensions,” in *FIRST INTERNATIONAL WORKSHOP ON CLOUD FOR BIO (C4Bio 2014)*, 2014, pp. 26–29 [Online]. Available: http://grids.ucs.indiana.edu/ptliupages/publications/PhylogeneticTreeDisplayWithClustering.pdf
[100] L. Stanberry, R. Higdon, W. Haynes, N. Kolker, W. Broomall, S. Ekanayake, A. Hughes, Y. Ruan, J. Qiu, E. Kolker, and Geoffrey Fox, “Visualizing the Protein Sequence Universe,” *Concurr. Comput.*, vol. 26, no. 6, pp. 1313–1325, Apr. 2014 [Online]. Available: http://grids.ucs.indiana.edu/ptliupages/publications/ecmls72i-stanberry.pdf
[101] Y. Liang, F. Wang, P. Zhang, J. H. Saltz, D. J. Brat, and J. Kong, “Development of a Framework for Large Scale Three-Dimensional Pathology and Biomarker Imaging and Spatial Analytics,” *AMIA Jt Summits Transl Sci Proc*, vol. 2017, pp. 75–84, Jul. 2017 [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/28815110
[102] P. Xuan, J. Denton, P. K. Srimani, R. Ge, and F. Luo, “Big Data Analytics on Traditional HPC Infrastructure Using Two-level Storage,” in *Proceedings of the 2015 International Workshop on Data-Intensive Scalable Computing Systems*, Austin, Texas, 2015, pp. 4:1–4:8 [Online]. Available: http://doi.acm.org/10.1145/2831244.2831253
[103] N. S. Islam, D. Shankar, X. Lu, M. Wasi-Ur-Rahman, and D. K. Panda, “Accelerating I/O Performance of Big Data Analytics on HPC Clusters through RDMA-Based Key-Value Store,” in *2015 44th International Conference on Parallel Processing*, 2015, pp. 280–289 [Online]. Available: http://dx.doi.org/10.1109/ICPP.2015.79
[104] L. Xu, Z. Huang, H. Jiang, L. Tian, and D. Swanson, “VSFS: A Versatile Searchable File System for HPC Analytics,” *https://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1133&context...*, 2013 [Online]. Available: https://digitalcommons.unl.edu/csetechreports/128/. [Accessed: 28-Sep-2018]
[105] F. Beneventi, A. Bartolini, C. Cavazzoni, and L. Benini, “Continuous learning of HPC infrastructure models using big data analytics and in-memory processing tools,” in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, 2017, pp. 1038–1043 [Online]. Available: http://dx.doi.org/10.23919/DATE.2017.7927143
[106] X. Lu, F. Liang, B. Wang, L. Zha, and Z. Xu, *DataMPI: Extending MPI to Hadoop-Like Big Data Computing*. 2014 [Online]. Available: http://dx.doi.org/10.1109/IPDPS.2014.90
[107] N. S. Islam, M. W. Rahman, J. Jose, R. Rajachandrasekar, H. Wang, H. Subramoni, C. Murthy, and D. K. Panda, “High performance RDMA-based design of HDFS over InfiniBand,” in *SC ’12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 2012, pp. 1–12 [Online]. Available: http://dx.doi.org/10.1109/SC.2012.65
[108] X. Lu, M. W. U. Rahman, N. Islam, D. Shankar, and D. K. Panda, “Accelerating Spark with RDMA for Big Data Processing: Early Experiences,” in *2014 IEEE 22nd Annual Symposium on High-Performance Interconnects*, 2014, pp. 9–16 [Online]. Available: http://dx.doi.org/10.1109/HOTI.2014.15
[109] S. Kamburugamuve, K. Ramasamy, M. Swany, and G. Fox, “Low Latency Stream Processing: Apache Heron with Infiniband & Intel Omni-Path,” in *Proceedings of the10th International Conference on Utility and Cloud Computing*, Austin, Texas, USA, 2017, pp. 101–110 [Online]. Available: http://dsc.soic.indiana.edu/publications/Heron_Infiniband.pdf,
[110] C. Barthels, I. Müller, T. Schneider, G. Alonso, and T. Hoefler, “Distributed Join Algorithms on Thousands of Cores,” *Proceedings VLDB Endowment*, vol. 10, no. 5, pp. 517–528, Jan. 2017 [Online]. Available: https://doi.org/10.14778/3055540.3055545
[111] C. Barthels, S. Loesing, G. Alonso, and D. Kossmann, “Rack-Scale In-Memory Join Processing Using RDMA,” in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, Melbourne, Victoria, Australia, 2015, pp. 1463–1475 [Online]. Available: http://doi.acm.org/10.1145/2723372.2750547
[112] B. L. Morris and A. Skjellum, “MPIgnite: An MPI-Like Language and Prototype Implementation

for Apache Spark," *arXiv [cs.DC]*, 15-Jul-2017 [Online]. Available: http://arxiv.org/abs/1707.04788
[113] "Graph 500 Benchmark." [Online]. Available: http://www.graph500.org. [Accessed: 21-Sep-2018]
[114] D. Chakrabarti and C. Faloutsos, "Graph Mining: Laws, Generators, and Algorithms," *ACM Comput. Surv.*, vol. 38, no. 1, Jun. 2006 [Online]. Available: http://doi.acm.org/10.1145/1132952.1132954
[115] A. L. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, Oct. 1999 [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/10521342
[116] T. Kolda, A. Pinar, T. Plantenga, and C. Seshadhri, "A Scalable Generative Graph Model with Community Structure," *SIAM J. Sci. Comput.*, vol. 36, no. 5, pp. C424–C452, Jan. 2014 [Online]. Available: https://doi.org/10.1137/130914218
[117] Maksudul Alam, Maleq Khan, Kalyan S. Perumalla, Madhav Marathe., "Generating massive scale-free networks: Novel parallel algorithms using the preferential attachment model," *ACM Transactions on Parallel Computing (submitted)*.
[118] Siddharth Suri and Sergei Vassilvitskii, "Counting triangles and the curse of the last reducer," in *Proceedings of the 20th international conference on World wide web*, 2011, pp. 607–614 [Online]. Available: http://dx.doi.org/10.1145/1963405.1963491
[119] M. Rahman and M. A. Hasan, "Approximate triangle counting algorithms on multi-cores," in *2013 IEEE International Conference on Big Data*, 2013, pp. 127–133 [Online]. Available: http://dx.doi.org/10.1109/BigData.2013.6691744
[120] S. Arifuzzaman, M. Khan, and M. Marathe, "A fast parallel algorithm for counting triangles in graphs using dynamic load balancing," in *2015 IEEE International Conference on Big Data (Big Data)*, 2015, pp. 1839–1847 [Online]. Available: http://dx.doi.org/10.1109/BigData.2015.7363957
[121] C. Seshadhri, A. Pinar, and T. Kolda, "Triadic Measures on Graphs: The Power of Wedge Sampling," in *Proceedings of the 2013 SIAM International Conference on Data Mining*, Society for Industrial and Applied Mathematics, 2013, pp. 10–18 [Online]. Available: https://doi.org/10.1137/1.9781611972832.2
[122] N. Alon, R. Yuster, and U. Zwick, "Color-coding," *J. ACM*, vol. 42, no. 4, pp. 844–856, 1995 [Online]. Available: http://dx.doi.org/10.1145/210332.210337
[123] Zhao Zhao, Guanying Wang, Ali R. Butt, Maleq Khan, V. S. Anil Kumar, and Madhav V. Marathe, "SAHAD: Subgraph Analysis in Massive Networks Using Hadoop," in *Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium*, 2012, pp. 390–401 [Online]. Available: http://dx.doi.org/10.1109/ipdps.2012.44
[124] G. M. Slota and K. Madduri, "Complex Network Analysis Using Parallel Approximate Motif Counting," in *2014 IEEE 28th International Parallel and Distributed Processing Symposium*, 2014, pp. 405–414 [Online]. Available: http://dx.doi.org/10.1109/IPDPS.2014.50
[125] L. Chen, B. Peng, S. Ossen, A. Vullikanti, M. Marathe, L. Jiang, and J. Qiu, "High-Performance Massive Subgraph Counting using Pipelined Adaptive-Group Communication," *arXiv [cs.DC]*, 25-Apr-2018 [Online]. Available: http://arxiv.org/abs/1804.09764
[126] T. Tu, C. A. Rendleman, D. W. Borhani, R. O. Dror, J. Gullingsrud, M. Ø. Jensen, J. L. Klepeis, P. Maragakis, P. Miller, K. A. Stafford, and D. E. Shaw, "A Scalable Parallel Framework for Analyzing Terascale Molecular Dynamics Simulation Trajectories," in *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, Austin, Texas, 2008, pp. 56:1–56:12 [Online]. Available: http://dl.acm.org/citation.cfm?id=1413370.1413427
[127] S. O. Yesylevskyy, "Pteros 2.0: Evolution of the fast parallel molecular analysis library for C++ and python," *J. Comput. Chem.*, vol. 36, no. 19, pp. 1480–1488, Jul. 2015 [Online]. Available: http://dx.doi.org/10.1002/jcc.23943
[128] R. T. McGibbon, K. A. Beauchamp, M. P. Harrigan, C. Klein, J. M. Swails, C. X. Hernández, C. R. Schwantes, L.-P. Wang, T. J. Lane, and V. S. Pande, "MDTraj: A Modern Open Library for the Analysis of Molecular Dynamics Trajectories," *Biophys. J.*, vol. 109, no. 8, pp. 1528–1532, Oct. 2015 [Online]. Available: http://dx.doi.org/10.1016/j.bpj.2015.08.015
[129] K. Hinsen, E. Pellegrini, S. Stachura, and G. R. Kneller, "nMoldyn 3: using task farming for a parallel spectroscopy-oriented analysis of molecular dynamics simulations," *J. Comput. Chem.*, vol. 33, no. 25, pp. 2043–2048, Sep. 2012 [Online]. Available: http://dx.doi.org/10.1002/jcc.23035
[130] J. A. MacGregor, M. A. Fahnestock, G. A. Catania, J. D. Paden, S. Prasad Gogineni, S. K. Young, S. C. Rybarski, A. N. Mabrey, B. M. Wagman, and M. Morlighem, "Radiostratigraphy and age structure of the Greenland Ice Sheet," *J Geophys Res Earth Surf*, vol. 120, no. 2, pp. 212–241, Feb. 2015 [Online]. Available: http://dx.doi.org/10.1002/2014JF003215
[131] A. Ferro and L. Bruzzone, "Automatic Extraction and Analysis of Ice Layering in Radar Sounder Data," *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 3, pp. 1622–1634, Mar. 2013 [Online]. Available: http://dx.doi.org/10.1109/TGRS.2012.2206078

[132] A. Ilisei, A. Ferro, and L. Bruzzone, “A technique for the automatic estimation of ice thickness and bedrock properties from radar sounder data acquired at Antarctica,” in *2012 IEEE International Geoscience and Remote Sensing Symposium*, 2012, pp. 4457–4460 [Online]. Available: http://dx.doi.org/10.1109/IGARSS.2012.6350482

[133] G. J. Freeman, A. C. Bovik, and J. W. Holt, “Automated detection of near surface Martian ice layers in orbital radar data,” in *2010 IEEE Southwest Symposium on Image Analysis Interpretation (SSIAI)*, 2010, pp. 117–120 [Online]. Available: http://dx.doi.org/10.1109/SSIAI.2010.5483905

[134] M. Rahnemoonfar, G. C. Fox, M. Yari, and J. Paden, “Automatic Ice Surface and Bottom Boundaries Estimation in Radar Imagery Based on Level-Set Approach,” *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 9, pp. 5115–5122, 2017 [Online]. Available: http://dx.doi.org/10.1109/TGRS.2017.2702200

[135] J. E. Mitchell, D. J. Crandall, G. C. Fox, and J. D. Paden, “A semi-automatic approach for estimating near surface internal layers from snow radar imagery,” in *2013 IEEE International Geoscience and Remote Sensing Symposium - IGARSS*, 2013, pp. 4110–4113 [Online]. Available: http://dx.doi.org/10.1109/IGARSS.2013.6723737

[136] C. Panton, “Automated mapping of local layer slope and tracing of internal layers in radio echograms,” *Ann. Glaciol.*, vol. 55, no. 67, pp. 71–77, 2014 [Online]. Available: https://www.cambridge.org/core/journals/annals-of-glaciology/article/automated-mapping-of-local-layer-slope-and-tracing-of-internal-layers-in-radio-echograms/2277A291B7398A6BE8B3D1CC8536EF05. [Accessed: 30-Sep-2018]

[137] H. Kamangir, M. Rahnemoonfar, D. Dobbs, J. Paden, and F. G, “DEEP HYBRID WAVELET NETWORK FOR ICE BOUNDARY DETECTION IN RADAR IMAGERY,” in *International Geoscience and Remote Sensing Symposium, IGARSS 2018*, Valencia Spain, 2018 [Online]. Available: http://dsc.soic.indiana.edu/publications/deep-hybrid-wavelet.pdf

[138] J. B. A. Maintz and M. A. Viergever, “A survey of medical image registration,” *Med. Image Anal.*, vol. 2, no. 1, pp. 1–36, Mar. 1998 [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1361841501800268

[139] T. Mäkelä, P. Clarysse, O. Sipilä, N. Pauna, Q. C. Pham, T. Katila, and I. E. Magnin, “A review of cardiac image registration methods,” *IEEE Trans. Med. Imaging*, vol. 21, no. 9, pp. 1011–1021, Sep. 2002 [Online]. Available: http://dx.doi.org/10.1109/TMI.2002.804441

[140] C. R. Maurer and J. M. Fitzpatrick, “A Review of Medical Image Registration,” in *Interactive Image-Guided Neurosurgery*, American Association of Neurological Surgeons, Ed. 1993, pp. 17–44 [Online]. Available: http://www.cs.jhu.edu/~cis/cista/746/papers/Fitzpatrick93ReviewOfRegistration.pdf

[141] A. Gholipour, N. Kehtarnavaz, R. Briggs, M. Devous, and K. Gopinath, “Brain functional localization: a survey of image registration techniques,” *IEEE Trans. Med. Imaging*, vol. 26, no. 4, pp. 427–451, Apr. 2007 [Online]. Available: http://dx.doi.org/10.1109/TMI.2007.892508

[142] H. Lester and S. R. Arridge, “A survey of hierarchical non-linear medical image registration,” *Pattern Recognit.*, vol. 32, no. 1, pp. 129–149, Jan. 1999 [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320398000958

[143] F. Fontana, A. Crovetto, M. Bergognoni, and A. M. Casali, “Multiresolution registration for volume reconstruction in microscopical applications,” in *Medical Imaging 1993: Image Processing*, 1993, vol. 1898, pp. 55–61 [Online]. Available: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/1898/0000/Multiresolution-registration-for-volume-reconstruction-in-microscopical-applications/10.1117/12.154546.full. [Accessed: 26-Sep-2018]

[144] D. S. Fritsch, S. M. Pizer, B. S. Morse, D. H. Eberly, and A. Liu, “The multiscale medial axis and its applications in image registration,” *Pattern Recognit. Lett.*, vol. 15, no. 5, pp. 445–452, May 1994 [Online]. Available: http://www.sciencedirect.com/science/article/pii/016786559490135X

[145] C. Studholme, D. L. Hill, and D. J. Hawkes, “Automated three-dimensional registration of magnetic resonance and positron emission tomography brain images by multiresolution optimization of voxel similarity measures,” *Med. Phys.*, vol. 24, no. 1, pp. 25–35, Jan. 1997 [Online]. Available: http://dx.doi.org/10.1118/1.598130

[146] S. Saalfeld, A. Cardona, V. Hartenstein, and P. Tomančák, “As-rigid-as-possible mosaicking and serial section registration of large ssTEM datasets,” *Bioinformatics*, vol. 26, no. 12, pp. i57–63, Jun. 2010 [Online]. Available: http://dx.doi.org/10.1093/bioinformatics/btq219

[147] J. Lotz, J. Berger, B. Müller, K. Breuhahn, N. Grabe, S. Heldmann, A. Homeyer, B. Lahrmann, H. Laue, J. Olesch, M. Schwier, O. Sedlaczek, and A. Warth, “Zooming in: high resolution 3D reconstruction of differently stained histological whole slide images,” in *Medical Imaging 2014: Digital Pathology*, 2014, vol. 9041, p. 904104 [Online]. Available: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/9041/904104/Zooming-in--high-resolution-3D-reconstruction-of-differently-stained/10.1117/12.2043381.full. [Accessed: 26-Sep-2018]

[148] J. Lotz, J. Olesch, B. Muller, T. Polzin, P. Galuschka, J. M. Lotz, S. Heldmann, H. Laue, M. Gonzalez-Vallinas, A. Warth, B. Lahrmann, N. Grabe, O. Sedlaczek, K. Breuhahn, and J. Modersitzki, "Patch-Based Nonlinear Image Registration for Gigapixel Whole Slide Images," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 9, pp. 1812–1819, Sep. 2016 [Online]. Available: http://dx.doi.org/10.1109/TBME.2015.2503122

[149] C.-W. Wang, E. Budiman Gosno, and Y.-S. Li, "Fully automatic and robust 3D registration of serial-section microscopic images," *Sci. Rep.*, vol. 5, p. 15051, Oct. 2015 [Online]. Available: http://dx.doi.org/10.1038/srep15051

[150] Y. Xu, J. Geoffrey Pickering, Z. Nong, E. Gibson, J.-M. Arpino, H. Yin, and A. D. Ward, "A Method for 3D Histopathology Reconstruction Supporting Mouse Microvasculature Analysis," *PLoS One*, vol. 10, no. 5, p. e0126817, May 2015 [Online]. Available: https://journals.plos.org/plosone/article/file?id=10.1371/journal.pone.0126817&type=printable. [Accessed: 26-Sep-2018]

[151] J. Pichat, M. Modat, T. Yousry, and S. Ourselin, "A multi-path approach to histology volume reconstruction," 2015 [Online]. Available: https://ieeexplore.ieee.org/iel7/7150573/7163789/07164108.pdf

[152] L. König and J. Rühaak, "A fast and accurate parallel algorithm for non-linear image registration using Normalized Gradient fields," in *2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI)*, 2014, pp. 580–583 [Online]. Available: http://dx.doi.org/10.1109/ISBI.2014.6867937

[153] Y. Song, D. Treanor, A. J. Bulpitt, and D. R. Magee, "3D reconstruction of multiple stained histology images," *J. Pathol. Inform.*, vol. 4, no. 2, p. 7, Jan. 2013 [Online]. Available: http://www.jpathinformatics.org/article.asp?issn=2153-3539;year=2013;volume=4;issue=2;spage=7;epage=7;aulast=Song. [Accessed: 26-Sep-2018]

[154] Y. Song, D. Treanor, A. J. Bulpitt, N. Wijayathunga, N. Roberts, R. Wilcox, and D. R. Magee, "Unsupervised content classification based nonrigid registration of differently stained histology images," *IEEE Trans. Biomed. Eng.*, vol. 61, no. 1, pp. 96–108, Jan. 2014 [Online]. Available: http://dx.doi.org/10.1109/TBME.2013.2277777

[155] A. Bria, G. Iannello, L. Onofri, and H. Peng, "TeraFly: real-time three-dimensional visualization and annotation of terabytes of multidimensional volumetric images," *Nat. Methods*, vol. 13, p. 192, Feb. 2016 [Online]. Available: http://dx.doi.org/10.1038/nmeth.3767

[156] H. Peng, M. Hawrylycz, J. Roskams, S. Hill, N. Spruston, E. Meijering, and G. A. Ascoli, "BigNeuron: Large-Scale 3D Neuron Reconstruction from Optical Microscopy Images," *Neuron*, vol. 87, no. 2, pp. 252–256, Jul. 2015 [Online]. Available: https://www.cell.com/neuron/abstract/S0896-6273(15)00599-1. [Accessed: 26-Sep-2018]

[157] "3DHISTECH The Digital Pathology Company." [Online]. Available: http://www.3dhistech.com/. [Accessed: 19-Sep-2018]

[158] "MicroDimensionw with SLIDEMATCH AND VOLOOM products." [Online]. Available: https://micro-dimensions.com/. [Accessed: 19-Sep-2018]

[159] L. Fónyad, K. Shinoda, E. A. Farkash, M. Groher, D. P. Sebastian, A. M. Szász, R. B. Colvin, and Y. Yagi, "3-dimensional digital reconstruction of the murine coronary system for the evaluation of chronic allograft vasculopathy," *Diagn. Pathol.*, vol. 10, p. 16, Mar. 2015 [Online]. Available: http://dx.doi.org/10.1186/s13000-015-0248-6

[160] M. Feuerstein, H. Heibel, J. Gardiazabal, N. Navab, and M. Groher, "Reconstruction of 3-D Histology Images by Simultaneous Deformable Registration," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2011*, 2011, pp. 582–589 [Online]. Available: http://dx.doi.org/10.1007/978-3-642-23629-7_71

[161] "Oracle Spatial and Graph." [Online]. Available: https://www.oracle.com/database/spatial. [Accessed: 19-Sep-2018]

[162] "MapInfo Discover 3D." [Online]. Available: http://www.pitneybowes.com/us/location-intelligence/geographic-information-systems/mapinfo-discover-3d.html. [Accessed: 19-Sep-2018]

[163] "ESRI 3D GIS." [Online]. Available: http://www.esri.com/products/arcgis-capabilities/3d-gis. [Accessed: 19-Sep-2018]

[164] A. Eldawy and M. F. Mokbel, "SpatialHadoop: A MapReduce framework for spatial data," in *2015 IEEE 31st International Conference on Data Engineering*, 2015, pp. 1352–1363 [Online]. Available: http://dx.doi.org/10.1109/ICDE.2015.7113382

[165] J. Yu, J. Wu, and M. Sarwat, "GeoSpark: A Cluster Computing Framework for Processing Large-scale Spatial Data," in *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, Seattle, Washington, 2015, pp. 70:1–70:4 [Online]. Available: http://doi.acm.org/10.1145/2820783.2820860

[166] D. Xie, F. Li, B. Yao, G. Li, L. Zhou, and M. Guo, "Simba: Efficient In-Memory Spatial Analytics," in *Proceedings of the 2016 International Conference on Management of Data*, 2016, pp.

1071–1085 [Online]. Available: https://dl.acm.org/citation.cfm?id=2882903.2915237. [Accessed: 26-Sep-2018]
[167] S. You, J. Zhang, and L. Gruenwald, "Large-scale spatial join query processing in Cloud," in *2015 31st IEEE International Conference on Data Engineering Workshops*, 2015, pp. 34–41 [Online]. Available: http://dx.doi.org/10.1109/ICDEW.2015.7129541
[168] M. Tang, Y. Yu, Q. M. Malluhi, M. Ouzzani, and W. G. Aref, "LocationSpark: A Distributed In-Memory Data Management System for Big Spatial Data∗" [Online]. Available: http://www.vldb.org/pvldb/vol9/p1565-tang.pdf
[169] "Big Data and Extreme-scale Computing (BDEC) community website." [Online]. Available: https://www.exascale.org/bdec/. [Accessed: 30-Sep-2018]
[170] M. Asch, T. Moore, R. Badia, M. Beck, P. Beckman, T. Bidot, F. Bodin, F. Cappello, A. Choudhary, B. de Supinski, and Others, "Big data and extreme-scale computing: Pathways to Convergence-Toward a shaping strategy for a future software and data ecosystem for scientific inquiry," *Int. J. High Perform. Comput. Appl.*, vol. 32, no. 4, pp. 435–479, 2018 [Online]. Available: http://journals.sagepub.com/doi/abs/10.1177/1094342018778123?casa_token=sT2beKDQVrUAAAAA:1uRgQj6TPgkIVQGDmhZis0UPX3HBcnNaX1idNX-J2HYTI885xCW7iBSraGGXgDjMCAw1iy2H6VQ
[171] Geoffrey Fox, Judy Qiu, Shantenu Jha, Saliya Ekanayake, and Supun Kamburugamuve, "White Paper: Big Data, Simulations and HPC Convergence," in *BDEC Frankfurt workshop*, 2016 [Online]. Available: http://dsc.soic.indiana.edu/publications/HPCBigDataConvergence.Summary_IURutgers.pdf
[172] Geoffrey C.Fox, Shantenu Jha, Judy Qiu, and Andre Luckow, "Ogres: A Systematic Approach to Big Data Benchmarks," in *Big Data and Extreme-scale Computing (BDEC)*, 2015 [Online]. Available: http://www.exascale.org/bdec/sites/www.exascale.org.bdec/files/whitepapers/OgreFacets.pdf
[173] Geoffrey Fox, Judy Qiu, and Shantenu Jha, "High Performance High Functionality Big Data Software Stack," in *Big Data and Extreme-scale Computing (BDEC)*, 2014 [Online]. Available: http://www.exascale.org/bdec/sites/www.exascale.org.bdec/files/whitepapers/fox.pdf
[174] "NSF1849625 workshop series BDEC2: Toward a common digital continuum platform for big data and extreme-scale computing (BDEC2)." [Online]. Available: https://nsf.gov/awardsearch/showAward?AWD_ID=1849625&HistoricalAwards=false, https://www.exascale.org/bdec/. [Accessed: 30-Sep-2018]
[175] "Agenda for BDEC2 meeting at Indiana with multimedia annotations." [Online]. Available: https://docs.google.com/document/d/1Gipxn2YKckjDHyPG2tNKqSeenEL7AxX6cB43YPtCpQA/edit?usp=sharing. [Accessed: 22-Feb-2019]
[176] G. Fox, "Twister2 Tutorial at BigDat2019 Cambridge UK January 8-11 2019." [Online]. Available: https://twister2.gitbook.io/twister2/tutorial. [Accessed: 30-Jan-2019]
[177] G. C. Fox, V. Ishakian, V. Muthusamy, and A. Slominski, "Status of Serverless Computing and Function-as-a-Service(FaaS) in Industry and Research," *arXiv [cs.DC]*, 27-Aug-2017 [Online]. Available: http://arxiv.org/abs/1708.08028
[178] "Tensorflow: An open source machine learning framework for everyone." [Online]. Available: https://www.tensorflow.org/. [Accessed: 30-Sep-2018]
[179] Geoffrey Fox, Shantenu Jha, "The Promise of Learning Everywhere and MLforHPC," in *Online Resource for BDEC2 Second Meeting at Kobe Japan*, Kobe Japan [Online]. Available: http://dsc.soic.indiana.edu/presentations/JhaFox_BDEC2_Kobe_02-2019.pptx
[180] JCS Kadupitiya , Geoffrey C. Fox , and Vikram Jadhao, "Machine learning for performance enhancement of molecular dynamics simulations," presented at the International Conference on Computational Science ICCS2019 , Faro, Algarve, Portugal, 2018 [Online]. Available: http://dsc.soic.indiana.edu/publications/ICCS8.pdf
[181] JCS Kadupitiya, Geoffrey C. Fox, Vikram Jadhao, "Machine Learning for Parameter Auto-tuning in Molecular Dynamics Simulations: Efficient Dynamics of Ions near Polarizable Nanoparticles," Indiana University, Nov. 2018 [Online]. Available: http://dsc.soic.indiana.edu/publications/Manuscript.IJHPCA.Nov2018.pdf