A Cloud-based Brain Connectivity Analysis Tool

Laura Brattain[1], Mihnea Bulugioiu[1,2], Adam Brewster[1], Mark Hernandez[1], Heejin Choi[3], Taeyun Ku[3], Kwanghun Chung[3], and Vijay Gadepally[1]

[1]MIT Lincoln Laboratory, Lexington, MA USA
[2]Northeastern University, Boston, MA USA
[3]Institute of Medical Engineering and Science, Massachusetts Institute of Technology, Cambridge, MA USA

Abstract*—With advances in high throughput brain imaging at the cellular and sub-cellular level, there is growing demand for platforms that can support high performance, large-scale brain data processing and analysis. In this paper, we present a novel pipeline that combines Accumulo, D4M, geohashing, and parallel programming to manage large-scale neuron connectivity graphs in a cloud environment. Our brain connectivity graph is represented using vertices (fiber start/end nodes), edges (fiber tracks), and the 3D coordinates of the fiber tracks. For optimal performance, we take the hybrid approach of storing vertices and edges in Accumulo and saving the fiber track 3D coordinates in flat files. Accumulo database operations offer low latency on sparse queries while flat files offer high throughput for storing, querying, and analyzing bulk data. We evaluated our pipeline by using 250 gigabytes of mouse neuron connectivity data. Benchmarking experiments on retrieving vertices and edges from Accumulo demonstrate that we can achieve 1-2 orders of magnitude speedup in retrieval time when compared to the same operation from traditional flat files. The implementation of graph analytics such as Breadth First Search using Accumulo and D4M offers consistent good performance regardless of data size and density, thus is scalable to very large dataset. Indexing of neuron subvolumes is simple and logical with geohashing-based binary tree encoding. This hybrid data management backend is used to drive an interactive web-based 3D graphical user interface, where users can examine the 3D connectivity map in a Google Map-like viewer. Our pipeline is scalable and extensible to other data modalities.

I. Introduction

Top on the list of the US Government's BRAIN Initiative is the ability to map the human brain at different scales with improved throughput and resolutions [1]. A complete picture of the brain structure will provide new insights into how the human brain functions and may facilitate new treatments and drug discovery for brain disorders. Recent advances in intact brain imaging, such as the CLARITY [2] and MAP (Magnified Analysis of the Proteome) [3] tissue clearing techniques, make it possible to collect large volumetric images of brain tissue at cellular and sub-cellular resolutions. The high throughput and high resolution brain imagery, however, poses a challenge for efficient processing and analysis.

This material is based upon work supported by the Assistant Secretary of Defense for Research and Engineering under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Assistant Secretary of Defense for Research and Engineering.

We have developed an automated dense axonal fiber tracing pipeline that can track long-range fibers and construct 3D connectivity maps, which include not only the vertices and edges of a network graph, but also the 3D location information associated with each fiber track (Fig. 1). In addition to typical graph analysis, we are interested in identifying long-range neuron fiber connections and fiber crossings, both of which could reveal informative patterns when analyzed at cellular resolution and at long range ($\geq$ 1 mm).

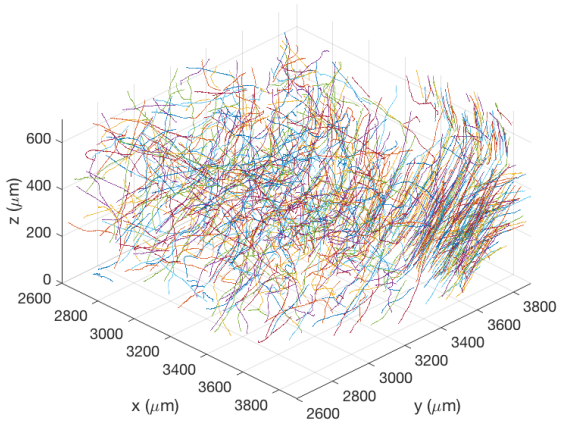


Fig. 1: Example dense neuron axonal fiber tracks

Brain graphs offer a framework to represent the structural or functional topology at multiple levels [4]–[6]. A number of software tools exist for analyzing topology of brain networks using graph theory [7]–[10]. They primarily focus on studying the correlations of anatomically separated brain regions. Few are designed for high throughput dense and long-range neuron analysis at the cellular level, which is critical for understanding brain circuits and for comparing healthy and diseased brains. High throughput and low latency analysis of brain data will require high speed databases and programming interfaces amenable to large scale graph analytics.

In this paper, we present an approach that combines Accumulo, D4M, and parallel programming to manage the large-scale brain connectivity graphs in a high performance computing environment [11], [12]. The Apache Accumulo database is based on Google's Bigtable database [13] and allows for fast

978-1-5386-3472-1/17/$31.00 ©2017 IEEE

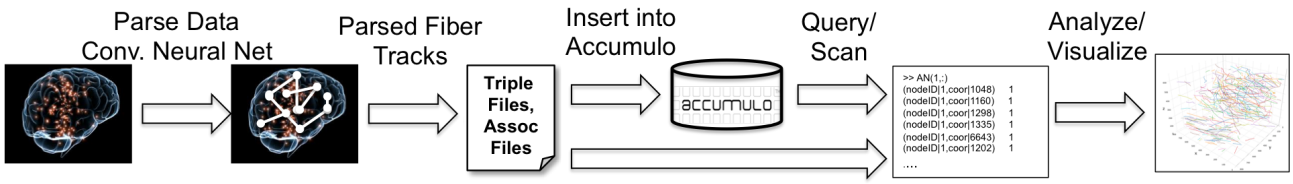


Fig. 2: Large-scale brain imagery processing and analysis pipeline

ingestion and extraction along with the capacity to hold large-scale data [14]. The Dynamic Distributed Dimensional Data Model (D4M) [15], [16] provides a mathematical framework, based on the mathematics of associative arrays [17], [18], that has been applied to diverse domains such as cyber security [19], bioinformatics, free text, and social media data. D4M can also be used to perform linear algebraic operations inside a database [20].

D4M works seamlessly with the pMatlab [21] (http://www.ll.mit.edu/pMatlab) parallel computing environment that allows users of MATLAB/Octave programming languages to use a single-program-multiple-data (SPMD) parallel programming model. pMatlab sits on top of a message passing interface [22] library such as MatlabMPI [23] or bcMPI [24] and has been been used for a variety of parallel computing applications [25]. For our application, the D4M-pMatlab connection is used to leverage parallel computing techniques in addition to the cloud-based big data techniques provided by D4M.

In this article, we extend the D4M-Accumulo interface to neural connectivity graphs computed from axon fiber imagery. This approach makes it possible to extract network vertices and edges with low latency, and perform fast and meaningful graph analytics by applying simple matrix operations.

We begin with a description of the proposed platform and the backend technologies including Accumulo, D4M, and binary tree encoding, followed by experimental results and performance benchmarking analysis using 250 gigabytes (GB) of mouse neuron data. We conclude with a summary and discussion of future work.

II. Methods

In this section, we describe our overall pipeline, as shown in Figure 2. The pipeline starts with an imaging processing module, which computes network connectivity graphs that contain vertices, edges, and 3D coordinates of fiber tracks. The graphs are then parsed into an associate array format. Vertices and edges are saved into an Accumulo database while 3D coordinates are saved to flat files. A real-time web-based 3D user interface displays the fiber tracks by querying data from Accumulo and flat files.

A. Technologies and Platform Design

1) Imaging Pipeline: The first step of the pipeline is to divide the original large data into subvolumes and apply image processing algorithms in parallel using pMatlab. The main image processing algorithms include a convolutional neural network (CNN) [26] for segmenting the axon fibers, and a number of morphological operations [27] for extracting fiber tracks, from which a network graph of vertices, edges, and 3D coordinates of the fiber tracks are computed. Our CNN has three convolutional and two fully connected layers. The last layer uses sigmoid activation, and all other layers use rectified linear units (ReLU) [28]. In the graphs, vertices correspond to the start/end of the fiber tracks and edges represent the connections (fiber tracks) between the vertices. These parsed graphs are then converted into an associative array format, which is amenable for graph analysis.

2) Accumulo and D4M: After the graphs are converted to an associative array format, they can be inserted into Accumulo or saved into flat files. Based on our benchtop experiments, we determined that a hybrid approach that leverages both Accumulo database and filesystem offers the optimal performance. We store the fiber graph vertices and edges in Accumulo while saving the fiber track 3D coordinates in flat files. 3D coordinates are much denser data when compared to the vertices and edges, and would slow down the retrieval time significantly if stored in Accumulo. In addition, since 3D coordinates are primarily used for morphology analysis and visualization, they are not accessed as frequently as vertices and edges. This hybrid configuration ensures low latency and high bandwidth access to data as needed for analytics and visualization.

3) Geohashing-based binary tree endcoding for subvolume indexing: The geohash was invented by Gustavo Niemeyer, and is a geocoding system based on a hierarchical spatial data structure which subdivides space into a grid that can be easily identified [29]. When used in a database, the structure of geohashed data has two advantages. First, data indexed by geohash will include all the points for a given rectangular area in contiguous slices. This is especially useful in a database system such as Accumulo where queries on a single index are much easier or faster than multiple-index queries. Second, this index structure can be used for a quick proximity search because the closest points are often among the closest geohashes [30].

One challenge presented by manipulating large brain data is that it is often difficult to drill down to any specific region of interest without having to go through lengthy and time consuming indexing process. To overcome this, we have designed a binary tree indexing schema based on geohashing. This allows us to identify specific neuron fibers in a particular subvolume easily and quickly. Figure 3 is an example of the

978-1-5386-3472-1/17/$31.00 ©2017 IEEE

TABLE I: Node Data Example

	nodeID|12-010110	subvolNum|010110	endYes|1	endNode|#	links|14	coor|49495112
nodeID|12-010110	1	1	1	1	1	1
nodeID|5-010110	0	1	1	0	0	0

TABLE II: Node Data Schema

	nodeID|#-bin	subvolNum|bin	endYes|#	endNode|#	links|#	coor|#
nodeID|#-bin	1	1	1	1	1	1

TABLE III: Link Data Example

	linkID|27-010110	subvolNum|010110	node1|36	node2|42	length|389
linkID|27-010110	1	1	1	1	1
linkID|56-100000	0	0	0	0	0

TABLE IV: Link Data Schema

	linkID|#-bin	subvolNum|bin	node1|#	node2|#	length|#
linkID|#-bin	1	1	1	1	1

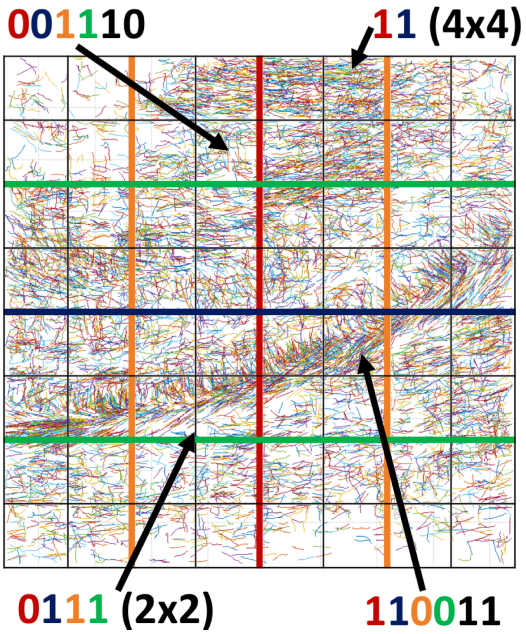


Fig. 3: Geohashing-based binary tree encoding

binary tree encoding scheme we designed. In this case, we use a string of 6 binary code to represent the entire dataset, with each digit representing a binary division. This results in 3 zoom levels. To illustrate the binary division, in Fig. 3, the color of each digit corresponds to the color of a vertical or horizontal division line.

Depending on the level of detail needed, binary codes can be easily made more specific by increasing the number of digits, resulting in more layers and finer subvolumes. This scalable schema is particularly useful for visualizing large datasets because it allows for easy query of data segments at multiple zoom levels, thus creating a Google Map like view.

4) Data Schema: In the following representation, nodes refer to graph vertices and links refers to edges connecting nodes in the graph.

The node associative array (Table II) contains the information for identifying a vertex globally. The `nodeID` is the locally unique number in each subvolume for each node. The subvolume number is the binary code where the node is located. `endYes` is 0 or 1 depending on whether or not the node is an end node, with 1 meaning it is the start or end of a fiber track, and 0 meaning it is a junction that connects to multiple other nodes. `endNode` is the `nodeID` of the other nodes that this node is connected to. `Links` is the `linkIDs` of the links that are connected to this node. `Coor` is the linear index of the coordinates of the node within the subvolume.

The link associative array (Table IV) contains the information about links except the 3D coordinates, which are saved in flat files in order to maintain fast query speed. The `linkID` and `subvolNum` are the same as `nodeID` and `subvolNum` in the node associative array. `node1` is the `nodeID` of one end of the link, while `node2` is the `nodeID` of the other. `length` contains the number of coordinates in the link.

B. Graph Analytics

In graph analytics, a commonly used algorithm is Breadth First Search (BFS) [18]. The BFS algorithm is used to traverse a graph data structure and find vertices connected to one another. In our application, BFS intuitively tells us which neurons are connected to other neurons. Using the associative array representation described previously, we can apply BFS to a node (or set of nodes) directly by multiplying two associative arrays [20].

To apply BFS on a node, we first extract the subvolume of interest from Accumulo to create an incidence matrix E. Then, the desired node is added to an associative array and multiplied by the incidence matrix of the subvolume's corresponding subgraph. This results in an associative array that has information about the nodes connected to the input nodes. The desired link information about the nodes is then extracted. This process can be repeated to find all nodes connected to this resultant. A snippet of the code used to perform BFS is shown below.

978-1-5386-3472-1/17/$31.00 ©2017 IEEE

In this example, we are looking for all nodes connected to node five in subvolume `01010101`. The subvolume ID `01010101` corresponds to the region of interest as defined by the binary tree encoding process described previously.

```
sv=tNodes(:,StartsWith('subvolNum|01010101,'));
nodeAssoc=tNodes(Row(sv),:);
E=nodeAssoc.'*nodeAssoc;
Result=E.'*Assoc('nodeID|5,','nodeID|5,',1);
Result(StartsWith('endNode|,'),:)
```

III. Experimental Results

A. Description of Dataset

The dataset used in this paper consists of microscopy images of axonal fibers from part of a mouse cerebral cortex. A Thy1-eGFP-M mouse was sacrificed according to the protocol approved by the MIT Institutional Animal Care and Use Committee and the Division of Comparative Medicine. A 170-μm-thick brain slice was obtained and processed by MAP protocol. The slice was stained by mouse SMI-312 antibody and anti-mouse IgG-Alexa Fluor 594 antibody to label all axonal fibers. After fourfold linear expansion in deionized water, the sample was imaged by the Leica TCS SP8 microscope system with a 25x, 0.95 NA water-immersion objective and a white light laser source. The acquired image volume consists of 800 slices of 2D tiff images that are vertically registered along the z axis, with a pixel resolution of 0.325 μm in x, y and 1 μm in z.

We divided the original 250 GB data into 50 x 50 sub-volumes using the binary tree encoding described above. The subvolumes were then processed in parallel using pMatlab. For segmenting the fibers, we used a CNN with three convolutional and two fully connected layers. Convolutions are 3 x 3 with stride and padding such that the total receptive field is 19 x 19 x 19. The computed vertices and edges were subsequently inserted into Accumulo, while the fiber track 3D coordinates were saved to flat files.

B. Query Performance

For illustration purposes, a subset of 16 x 16 subvolumes were used for testing. It included a total of 618,976 nodes. A randomly chosen single node from one random subvolume was extracted using both flat files and Accumulo. Performance time was measured as the average of 5 retrievals. Notice the y axis scale is logarithmic to better show the trend that Accumulo outperforms flat files at every subvolume density (Fig. 4). In this test data, Accumulo offers an average of 1-2 orders of magnitude of speedup. Performance of using flat files is limited by the fact that in order to extract one node from one subvolume, the entire subvolume must be loaded. Retrieval rate is relatively constant in Accumulo regardless of subvolume density, while flat file performance quickly degrades as density increases. The fast query performance of Accumulo is desirable because it allows for quick extraction of useful information for graph analytics.

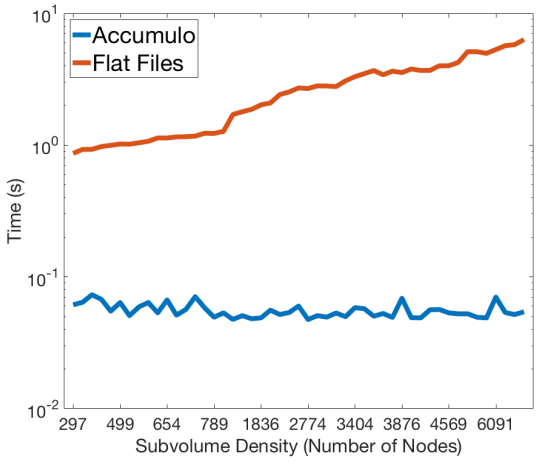


Fig. 4: Comparison of node query time in Accumulo and flat files

C. Graph Analytics

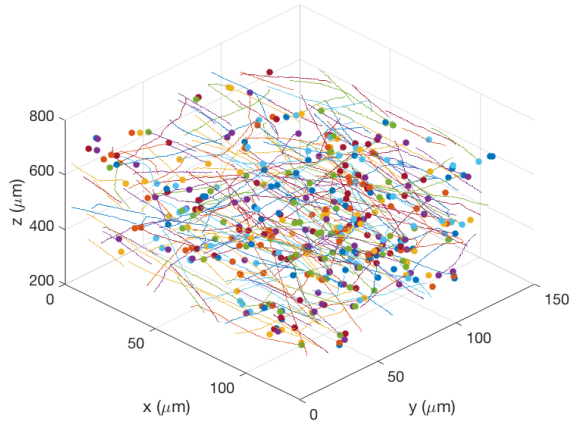
As described previously, one important application of our system is to perform graph analytics on neuron axon fibers in order to gain insights on brain structural patterns. There are numerous graph analytics of interest. With the combination of Accumulo and D4M, our system supports fast and easy iterative analysis with just a few lines of code. For example, in a given region of interest, we often want to know which nodes have more connections. We can achieve this by using the BFS algorithm described earlier. Using the same 16 x 16 subvolume test data, extracting all nodes with three or more connections results in 383 nodes, and extracting nodes with four or more connections results in 32 nodes (Fig. 5). Based on our visual inspection, all the connections were correctly identified by the algorithm. To find all the nodes connected in some way to the start node takes 3.7 ± 0.5 seconds regardless of the subvolume density and size. The performance of the BFS algorithm is fast and consistent, thus scalable to any data size.

D. Interactive Web-based 3D Visualization

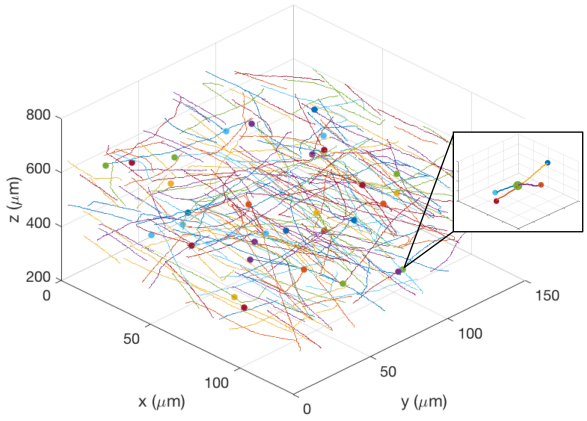
Leveraging the backend that combines geohashing-based binary tree encoding, Accumulo, and flat files, we implemented an interactive web-based 3D graphical user interface (GUI) in Python. With binary tree encoding, it is possible to visualize large 3D data at multiple zoom levels with low latency. Fig. 6 is an example of dividing a data volume into 16 x 16 subvolumes, and representing the division of data by an eight digit binary tree encoding, which produces four zoom levels. The inset shows an example of fusing 4 adjacent subvolumes extracted from the original volume for a zoomed-in visualization.

In the white matter of the brain regions, the density of the connectivity can quickly become very high. To maintain interactive nature of the GUI, we designed our visualization algorithm in a way that the total number of fibers shown is

978-1-5386-3472-1/17/$31.00 ©2017 IEEE



(a) Nodes with 3 or more connections



(b) Nodes with 4 or more connections

Fig. 5: Example of BFS for finding nodes with multiple connections

constant, regardless of density. This ensures that our GUI will run smoothly independent of differences in graphic engine capabilities.

IV. Conclusion and Future Work

In this paper, we described a cloud-based system for performing large-scale brain connectivity analysis using Apache Accumulo and D4M. We demonstrated that our approach can achieve fast data query and extraction for graph analytics and visualization. Indexing of subvolumes is simple and logical with geohashing-based binary tree encoding, which supports a Google Map-like viewer with multiple zoom levels. There are many avenues for future work. First, we would like to enhance the web GUI by making it more interactive and user friendly. Further, we intend to scale up to process much larger datasets (terabytes and above) with the goal of one day being able to perform such analysis on the human brain. We are currently exploring the use of Graphulo [31] to perform graph analytics directly inside Accumulo. We are also exploring the use of a polystore database such as BigDAWG [32] as a data

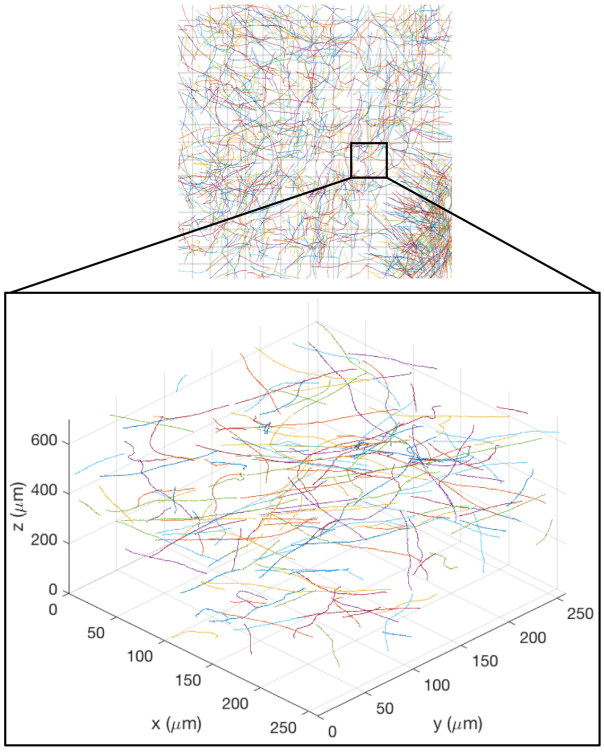


Fig. 6: Example of Google Map-like viewer showing the original data and a zoomed-in view of 4 neighboring subvolumes extracted using binary tree encoding

management solution that closely matches data sources with data management technology.

Acknowledgment

The authors wish to thank Brian Telfer, Siddharth Samsi, and Arjun Majumdar from MIT Lincoln Laboratory for their valuable inputs on this project. The authors also wish to thank the Lincoln Laboratory Supercomputing Center for their computational support.

References

[1] T. R. Insel, S. C. Landis, and F. S. Collins, "The nih brain initiative," *Science*, vol. 340, no. 6133, pp. 687–688, 2013.

[2] K. Chung and K. Deisseroth, "Clarity for mapping the nervous system," *Nat Meth*, vol. 10, no. 6, pp. 508–513, 06 2013. [Online]. Available: http://dx.doi.org/10.1038/nmeth.2481

[3] T. Ku, J. Swaney, J.-Y. Park, A. Albanese, E. Murray, J. H. Cho, Y.-G. Park, V. Mangena, J. Chen, and K. Chung, "Multiplexed and scalable super-resolution imaging of three-dimensional protein localization in size-adjustable tissues," vol. 34, no. 9, pp. 973–981. [Online]. Available: http://dx.doi.org/10.1038/nbt.3641

[4] O. Sporns, "The human connectome: a complex network: The human connectome," vol. 1224, no. 1, pp. 109–125. [Online]. Available: http://doi.wiley.com/10.1111/j.1749-6632.2010.05888.x

[5] S. M. Smith, K. L. Miller, G. Salimi-Khorshidi, M. Webster, C. F. Beckmann, T. E. Nichols, J. D. Ramsey, and M. W. Woolrich, "Network modelling methods for FMRI," vol. 54, no. 2, pp. 875–891.

[6] E. T. Bullmore and D. S. Bassett, "Brain graphs: graphical models of the human brain connectome," vol. 7, pp. 113–140.

[7] NITRC: GraphVar: A user-friendly toolbox for comprehensive graph analyses of functional brain connectivity: Tool/resource info. [Online]. Available: https://www.nitrc.org/projects/graphvar/

[8] Brain connectivity toolbox. [Online]. Available: https://sites.google.com/site/bctnet/

[9] S. M. H. Hosseini, F. Hoeft, and S. R. Kesler, "GAT: A graph-theoretical analysis toolbox for analyzing between-group differences in large-scale structural and functional brain networks," vol. 7, no. 7, p. e40709. [Online]. Available: http://dx.plos.org/10.1371/journal.pone.0040709

[10] B. He, Y. Dai, L. Astolfi, F. Babiloni, H. Yuan, and L. Yang, "eConnectome: A MATLAB toolbox for mapping and imaging of brain functional connectivity," vol. 195, no. 2, pp. 261–269. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0165027010006497

[11] A. Reuther, J. Kepner, W. Arcand, D. Bestor, B. Bergeron, C. Byun, M. Hubbell, P. Michaleas, J. Mullen, A. Prout *et al.*, "Llsupercloud: Sharing hpc systems for diverse rapid prototyping," in *High Performance Extreme Computing Conference (HPEC), 2013 IEEE*. IEEE, 2013, pp. 1–6.

[12] A. Prout, J. Kepner, P. Michaleas, W. Arcand, D. Bestor, B. Bergeron, C. Byun, L. Edwards, V. Gadepally, M. Hubbell *et al.*, "Enabling on-demand database computing with mit supercloud database management system," in *High Performance Extreme Computing Conference (HPEC), 2015 IEEE*. IEEE, 2015, pp. 1–6.

[13] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber, "Bigtable: A distributed storage system for structured data," *ACM Transactions on Computer Systems (TOCS)*, vol. 26, no. 2, p. 4, 2008.

[14] J. Kepner, W. Arcand, D. Bestor, B. Bergeron, C. Byun, V. Gadepally, M. Hubbell, P. Michaleas, J. Mullen, A. Prout, A. Reuther, A. Rosa, and C. Yee, "Achieving 100,000,000 database inserts per second using accumulo and d4m," in *High Performance Extreme Computing Conference (HPEC), 2014 IEEE*, Sept 2014, pp. 1–6.

[15] J. Kepner, C. Anderson, W. Arcand, D. Bestor, B. Bergeron, C. Byun, M. Hubbell, P. Michaleas, J. Mullen, D. O'Gwynn, A. Prout, A. Reuther, A. Rosa, and C. Yee, "D4m 2.0 schema: A general purpose high performance schema for the accumulo database," in *High Performance Extreme Computing Conference (HPEC), 2013 IEEE*, Sept 2013, pp. 1–6.

[16] V. Gadepally, J. Kepner, W. Arcand, D. Bestor, B. Bergeron, C. Byun, L. Edwards, M. Hubbell, P. Michaleas, J. Mullen, A. Prout, A. Rosa, C. Yee, and A. Reuther, "D4m: Bringing associative arrays to database engines," in *High Performance Extreme Computing Conference (HPEC), 2015 IEEE*, Sept 2015, pp. 1–6.

[17] J. Kepner and V. Gadepally, "Adjacency matrices, incidence matrices, database schemas, and associative arrays," in *International Parallel & Distributed Processing Symposium Workshops (IPDPSW)*, 2014.

[18] J. Kepner, V. Gadepally, D. Hutchison, H. Jananthan, T. Mattson, S. Samsi, and A. Reuther, "Associative array model of sql, nosql, and newsql databases," in *High Performance Extreme Computing Conference (HPEC), 2016 IEEE*. IEEE, 2016, pp. 1–9.

[19] T. Moyer and V. Gadepally, "High-throughput ingest of data provenance records into accumulo," in *High Performance Extreme Computing Conference (HPEC), 2016 IEEE*. IEEE, 2016, pp. 1–6.

[20] V. Gadepally, J. Bolewski, D. Hook, D. Hutchison, B. Miller, and J. Kepner, "Graphulo: Linear algebra graph kernels for nosql databases," in *Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International*. IEEE, 2015, pp. 822–830.

[21] N. Travinin Bliss and J. Kepner, "pmatlab parallel matlab library," *Int. J. High Perform. Comput. Appl.*, vol. 21, no. 3, pp. 336–359, Aug. 2007.

[22] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: portable parallel programming with the message-passing interface*. MIT press, 1999, vol. 1.

[23] J. Kepner and S. Ahalt, "Matlabmpi," *Journal of Parallel and Distributed Computing*, vol. 64, no. 8, pp. 997–1005, 2004.

[24] D. E. Hudak, N. Ludban, A. Krishnamurthy, V. Gadepally, S. Samsi, and J. Nehrbass, "A computational science ide for hpc systems: design and applications," *International journal of parallel programming*, vol. 37, no. 1, pp. 91–105, 2009.

[25] A. Krishnamurthy, S. Samsi, and V. Gadepally, "Parallel matlab techniques," *arXiv preprint arXiv:1407.2636*, 2014.

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[27] M. Kerschnitzki, P. Kollmannsberger, M. Burghammer, G. N. Duda, R. Weinkamer, W. Wagermaier, and P. Fratzl, "Architecture of the osteocyte network correlates with bone material quality: OSTEOCYTE NETWORK ARCHITECTURE CORRELATES WITH BONE MATERIAL QUALITY," vol. 28, no. 8, pp. 1837–1845. [Online]. Available: http://doi.wiley.com/10.1002/jbmr.1927

[28] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, J. Fürnkranz and T. Joachims, Eds. Omnipress, 2010, pp. 807–814. [Online]. Available: http://www.icml2010.org/papers/432.pdf

[29] Geohash - wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Geohash

[30] A. Fox, C. Eichelberger, J. Hughes, and S. Lyon, "Spatio-temporal indexing in non-relational distributed databases," in *Big Data, 2013 IEEE International Conference on*. IEEE, 2013, pp. 291–299.

[31] D. Hutchison, J. Kepner, V. Gadepally, and A. Fuchs, "Graphulo implementation of server-side sparse matrix multiply in the accumulo database," in *High Performance Extreme Computing Conference (HPEC), 2015 IEEE*. IEEE, 2015, pp. 1–7.

[32] V. Gadepally, P. Chen, J. Duggan, A. Elmore, B. Haynes, J. Kepner, S. Madden, T. Mattson, and M. Stonebraker, "The bigdawg polystore system and architecture," in *High Performance Extreme Computing Conference (HPEC), 2016 IEEE*. IEEE, 2016, pp. 1–6.

978-1-5386-3472-1/17/$31.00 ©2017 IEEE