Analyzing Large Data Sets from XGC1 Magnetic Fusion Simulations using Apache Spark

R. Michael Churchill
Princeton Plasma Physics Laboratory
Princeton, NJ, USA
rchurchi@pppl.gov

***Abstract*— Apache Spark is explored as a tool for analyzing large data sets from the magnetic fusion simulation code XGC1. Implementation details of Apache Spark on the NERSC Edison supercomputer are discussed, including binary file reading, and parameter setup. An unsupervised machine learning algorithm, k-means clustering, is applied to XGC1 particle distribution function data, showing that highly turbulent spatial regions do not have common coherent structures, but rather broad, ring-like structures in velocity space.**

Keywords—spark, magnetic fusion, simulation, machine-learning, distributed computing, k-means clustering

I. Introduction

Enabling researchers to perform data analysis and exploration on large scientific datasets is necessary to extract as much scientific understanding from their data as possible. This need is pressing for the big data sets generated from the magnetic fusion simulation code XGC1 [1,2], which generates saved datasets of order 100 TB per 24-hour simulation, running on leadership class supercomputers such as Titan at OLCF. As XGC1 implements very fundamental equations of motion, resulting in complex nonlinear behavior, analysis of XGC1 output is similar to analysis needed for experimental outputs, requiring exploration and reduction of data to discover the salient physics.

The Apache Spark project[3] provides a framework with several tools useful for working with large datasets: distribution of large datasets across multiple compute nodes, machine-learning and analytics library, and a simple Python interface. Exploratory tests on moderate sized (~100 GB) datasets were undertaken using Apache Spark installed on the Edison supercomputer at NERSC. Lessons learned will be presented, including working with scientific data files such as HDF5 and ADIOS, optimal Spark settings such as memory use per node and data partitions, and scalings for reading and operating on large datasets. Additionally, a k-means clustering technique was applied using Apache Spark to the XGC1 particle distribution functions data. Using this technique, similar regions of velocity space were discovered when strong turbulence in the plasma ("blobs") was present, indicating a similar generating mechanism for these features.

II. XGC1

The XGC1 simulation code is a multi-physics, multi-scale simulation code for magnetic fusion devices called tokamaks. It's aim is to include all of the physics necessary to simulate the entire tokamak, but focused on the difficult edge region, which plays an important role in fusion performance. XGC1 is a hybrid particle-in-cell (PIC) code using both particles and grid data during the simulation. High fidelity production simulations are run on the Titan supercomputer at Oak Ridge National Laboratory for best performance.

XGC1 moves particles according to equations of motions (dependent on the electric potential), and then uses those particles in solving a field equation for the electric potential. Writing out particle data for each timestep of the simulation is prohibitive due to size and time to write out. As in many other fields, the solution is to reduce the size of the dataset. Particle distribution functions, f, are formed, which are 5D arrays (3D in space, 2D in velocity), and can be further reduced by taking statistical moments of the particle distribution function, and averaging over specific spatial directions (e.g. averages over surfaces of constant magnetic flux).

However, many physics of interest may not be captured by the reduced moments, including wave-particle interactions. Analyzing particle distribution functions directly may provide more insight into the physics governing the highly complex edge. Yet analyzing particle distribution functions is made difficult by the size of the dataset, which is of order ~100 GB per simulation timestep. A solution which can handle large data sets and provide a familiar interface is highly desirable to reduce user code development time, while still allowing physicists to explore and interact with these large XGC1 datasets.

III. Apache Spark on Edison

Apache Spark was chosen as a solution to the problem of analyzing large XGC1 datasets, as it allows easy distribution of data sets for parallel processing, and provides an easy to use Python interface for rapid prototyping and building of analysis codes. The NERSC Edison supercomputer was chosen, as it was already setup with Apache Spark (including allowing TCP/IP communication necessary for current versions of Spark), and had large data transfer rates from the Titan supercomputer. Apache Spark v1.3.1 was used for this study.

Several obstacles had to be overcome in configuring Spark for this use case. First, Spark has a number of predefined data reading routines, but focused on text files, or data stored in file formats specific to the Hadoop ecosystem. The XGC1 particle distribution function data is stored in a binary file format, the ADIOS [4] *.bp file format (easily converted to HDF5 files). The solution to this problem was creating a Python read function, which takes as an argument an index range, opens a file handle to the ADIOS bp file, using the Python ADIOS package, then reads in the data for that index range. The Spark Resilient Distributed Dataset (RDD), Spark's distributed data abstraction, is populated then by first parallelizing the data indices to read across the compute partitions, then mapping these indices using the read function just described. Example code is shown in Figure 1. Data loading times on a single compute node (24 cores) were roughly ~1 GB/s for up to 33 GB.

Figure 1: Reading routine for binary file formats with Spark

```
import adios as ad
import numpy as np

def read(ind):
    f = ad.file('/path/to/file')
    tmp = f['data_name'][:,:,ind[0]:ind[-1]+1,:]
    f.close()
    return tmp

Nnodes = 10
NcoresPerNode = 22
Nparts = Nnodes*NcoresPerNode*4
indices = np.array_split(np.arange(0,Nrecords),Nparts)
index = sc.parallelize(indices,Nparts)

rdd = index.map(lambda v: read(v))
```

The second obstacle was in defining Spark parameters for use on Edison. Each compute node on Edison has 64 GB, of which 61 GB is typically available for user/application data. However, Spark defaults to allowing 54% of the total JVM heap size, so generally data stored in memory should not exceed about 32 GB. Before understanding this, several attempts to set executor memory to much larger (e.g. --executor-memory 48 GB) resulted in remote logoffs from the compute nodes, presumably due to memory pressure. Additionally, it was found for best performance to leave 2 cores per node unused by Spark (i.e. --executor-cores 22).

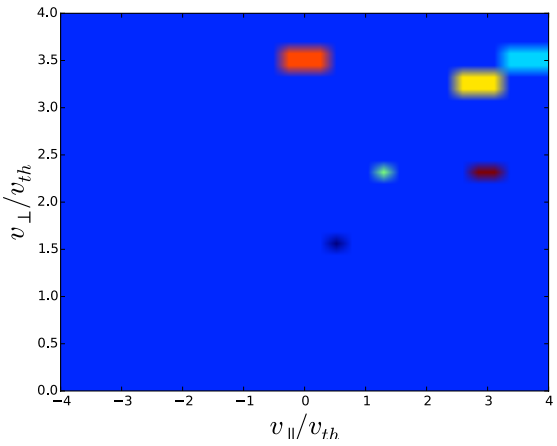
IV. K-means Clustering of XGC1 Data

Coherent structures in real space in the plasma density are experimentally observed in tokamaks, commonly referred to as "blobs" [5]. Concerns over the fast transport and interaction with the walls due to "blobs" have led to intense research and study of blobs, yet a full understanding of their origin remains elusive [6]. Some theories suggest velocity-space coherent structures play a role in the creation and sustaining of "blobs" [7,8], while others have pointed out that such structures may not be important [9].

Here we analyze a single timestep of XGC1 particle distribution function data, to determine common velocity signatures in regions with turbulent blobby activity across different spatial points. The algorithm we use is the k-means clustering algorithm, part of the MLlib library in Spark. Each spatial point in the simulation has a 2D velocity-space distribution function, which we will effectively stack against those of other spatial points, as if stacking multiple images. The k-means clustering algorithm is then applied to each velocity bin series, which cluster velocity bins which vary together. This is very similar to doing multiple correlation analysis between multiple velocity-space bins, across space.

Applying the k-means clustering algorithm to regions in the edge, $0.9 < \psi < 1.0$, a dataset approximately 60 GB in size, resulted in no definitive clustering of velocity bins, as seen in Figure 2, and evidenced by the clusters changing around for successive applications of k-means.

Figure 2: k-means clustering on entire edge region, showing no appreciable clusters in velocity space. Clusters formed vary from run to run, indicating noise.

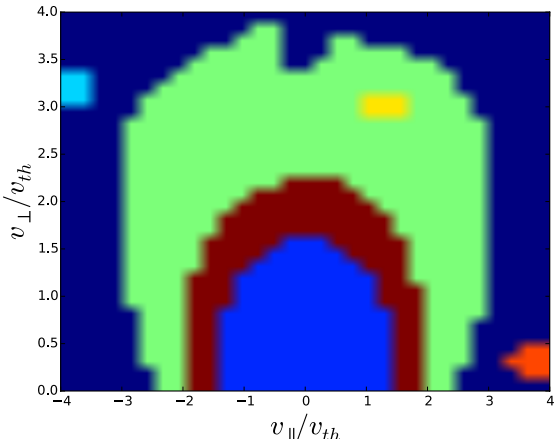


However, when applying the k-means clustering algorithm by stacking only edge spatial points which exhibited blobby turbulent behavior, a very distinct, ring-like structure in velocity-space was discovered, with three dominant clusters (see Figure 3). This indicates that in turbulent regions, there are successively increasing regions of particle speed (or energy) where common signatures are found in the particle distribution functions.

This doesn't reveal the physics governing blob creation, but does give a hint to physicists of mechanisms to explore. It seems to indicate that coherent structures do not form in common areas of velocity-space, contrary to some opinions, though we cannot rule out coherent structures at random velocities, which this analysis would not isolate.

V. Conclusion

Figure 3: k-means clustering stacking only spatial areas with strong turbulence. Ring structure of the clusters indicate similar generating mechanisms for blobs.



Apache Spark has been a very useful tool for enabling more detailed analysis of large data sets from the magnetic fusion code XGC1. We have successfully run Spark on the NERSC Edison supercomputer, reading in scientific binary data formats. A k-means clustering algorithm was applied to XGC1 particle distribution function data, resulting in clusters of velocity bins, revealing common signatures in blobby turbulence areas. Future work will increase the number of timesteps used, to cluster distribution function data across space and time.

Acknowledgment

This work was supported by the U.S. Department of Energy under Contract No. DE-AC02-09CH11466 and Grant No. DE-SC000801. This work used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

References

[1] Chang, C. S., Ku, S., Diamond, P., Adams, M., Barreto, R., Chen, et al. (2009). Whole-volume integrated gyrokinetic simulation of plasma turbulence in realistic diverted-tokamak geometry. *Journal of Physics: Conference Series*, *180*(1), 12057. http://doi.org/10.1088/1742-6596/180/1/012057

[2] Ku, S., Chang, C. S., & Diamond, P. H. (2009). Full-f gyrokinetic particle simulation of centrally heated global ITG turbulence from magnetic axis to edge pedestal top in a realistic tokamak geometry. *Nuclear Fusion*, *49*(11), 115021. http://doi.org/10.1088/0029-5515/49/11/115021

[3] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing* (HotCloud'10). USENIX Association, Berkeley, CA, USA, 10-10.

[4] Liu, Q., Logan, J., Tian, Y., Abbasi, H., Podhorszki, N., Choi, J. et al. (2014), Hello ADIOS: the challenges and lessons of developing leadership class I/O frameworks. Concurrency Computat.: Pract. Exper., 26: 1453–1473. doi:10.1002/cpe.3125

[5] Zweben, S. J., Boedo, J. A., Grulke, O., Hidalgo, C., LaBombard, B., Maqueda, R. J., … Terry, J. L. (2007). Edge turbulence measurements in toroidal fusion devices. *Plasma Physics and Controlled Fusion*, *49*(7), S1–S23. http://doi.org/10.1088/0741-3335/49/7/S01

[6] Krasheninnikov, S. I., Pigarov, A. Y., & Lee, W. (2015). Physics of the edge plasma and first wall in fusion devices: synergistic effects. *Plasma Physics and Controlled Fusion*, *57*(4), 44009. http://doi.org/10.1088/0741-3335/57/4/044009

[7] Kosuga, Y., & Diamond, P. H. (2012). Drift hole structure and dynamics with turbulence driven flows. *Physics of Plasmas*, *19*(7). http://doi.org/10.1063/1.4737197

[8] Dupree, T. H. (1972). Theory of Phase Space Density Granulation in Plasma. *Physics of Fluids*, *15*(2), 334. http://doi.org/10.1063/1.1693911

[9] Krommes, J. a. (1997). The clump lifetime revisited: Exact calculation of the second-order structure function for a model of forced, dissipative turbulence. *Phys. Plasmas*, *4655*(1997), 1342. http://doi.org/10.1063/1.872148