Using Thrill to Process Scientific Data on HPC

Mariia Karabin
Clemson University
mariiak@clemson.edu

Supreeth Suresh
University of Wyoming
ssuresh@uwyo.edu

Xinyu Chen
University of New Mexico
xychen@cs.unm.edu

Ivo Jimenez
University of California, Santa Cruz
ivo.jimenez@ucsc.edu

Li-Ta Lo
Los Alamos National Laboratory
ollie@lanl.gov

Pascal Grosset
Los Alamos National Laboratory
pascalgrosset@lanl.gov

ABSTRACT

With ongoing improvement of computational power and memory capacity, the volume of scientific data keeps growing. To gain insights from vast amounts of data, scientists are starting to look at Big Data processing tools like Apache Spark. In this poster, we explore Thrill, a framework for big data computation on HPC clusters that provides an interface similar to systems like Apache Spark but delivers higher performance. Using Thrill, we implemented several operations to analyze data from plasma physics and molecular dynamics simulations. Those operations were implemented with less programming effort than hand-crafted data processing programs. The preliminary analysis results have been verified for correctness by domain scientists at LANL.

CCS CONCEPTS

• Computing methodologies → MapReduce algorithms; **• Mathematics of computing → Cluster analysis**; **Exploratory data analysis**;

KEYWORDS

Distributed Data Processing, Clustering, Big Data, Molecular Dynamics

ACM Reference format:
Mariia Karabin, Supreeth Suresh, Xinyu Chen, Ivo Jimenez, Li-Ta Lo, and Pascal Grosset. 2018. Using Thrill to Process Scientific Data on HPC. In *Proceedings of , Dallas, TX, USA, November, 2018 (SC18),* 2 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The challenge of *Big Data* has expanded from traditional business intelligence to scientific research. Scientists need to put greater effort into processing and analyzing their data for scientific discovery. However, current state of the art practices are far from ideal. For example, at LANL, scientists studying molecular dynamics have to process thousands to millions of data files by serial scripts. Other scientists studying plasma physics have to develop and maintain parallel MPI programs to analyze their data with great effort. Related works like [4, 5] showed, it would greatly improve scientists' productivity if a general purpose, high level, distributed parallel data processing framework is available.

Thrill[3] is a recently developed parallel data processing C++ library that aims to introduce high level programming interfaces and declarative programming style into high performance computing communities to facilitate data processing and analysis. The declarative style enables the Thrill library to optimize parallel operations by pipelining and chaining operations.It also introduces a more succinct programming style similar to Apache Spark and other data flow languages.

The goal of this poster is to explore how a declarative language paradigm can be used to solve big data scientific problems and how this approach can be more efficient (in terms of less time spent coding and debugging) than creating ad hoc analysis programs. Using the Thrill library, we implemented time series analysis for Accelerated Molecular Dynamics (AMD) simulations[2], K-means clustering of AMD states, and particle trajectories for plasma physics simulations [1]. The correctness of our results were verified by the scientists. The scaling studies show that to some extent, our implementations also have a strong scalability.

2 METHODS

The Thrill library distributes data into distributed immutable arrays (DIAs). It provides scalable algorithmic primitives such as *Map, ReduceByKey, Sort* and *Join* to operate on DIAs. Essentially, DIAs are equivalent to tables in relational databases and algorithmic primitives are equivalent to SQL and data flow (Spark) operations.

2.1 Molecular Dynamics

2.1.1 Query of the Time Series. Scientists provided us with millions of state-files from the Accelerated Molecular Dynamics (AMD) simulations[2]. Each state file contains coordinates of hundreds of atoms. AMD simulations also produce trajectory files to describe the sequences and durations of each state.

The steps to query the time series of potential energy in Thrill are as follows: we first *Mapped* a function that uses the LAMMPS library to calculate the total potential energy (PE) per state to each state file in order to get a DIA of {state_id, potential_energy}. Similarly we used *Map* and *PrefixSum* to create a DIA of {state_id, time} from a trajectory file. Then we used *Join* to get the final time series of potential energies (PEs). Figure 1 illustrates the entire flow of operations to query the time series.

[1]The publication has been assigned the LANL identifier LA-UR-18-28182.

SC18, November, 2018, Dallas, TX, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

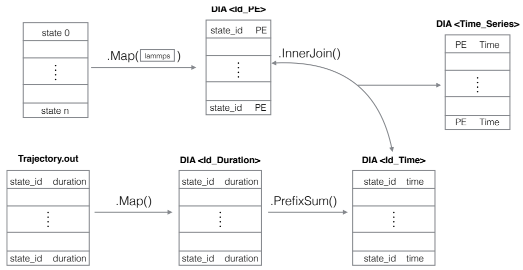


Figure 1: Querying the time series of the total potential energies per state.

2.1.2 K-means clustering of AMD states. To find out clusters of similar AMD states, we *Mapped* a function that uses LAMMPS to calculate common neighbor analysis (CNA) features for each state. This operation created a DIA of states characterized by a 6-dimensional CNA-feature vector {state_id, cna_1, cna_2, cna_3, cna_4, cna_5, cna_6}. By sampling k random centroids initially, we iteratively applied *Map* and *ReduceByKey* to the above DIA of CNA-feature vectors to update the centroids and members of every clusters later.

2.2 Particle Simulation

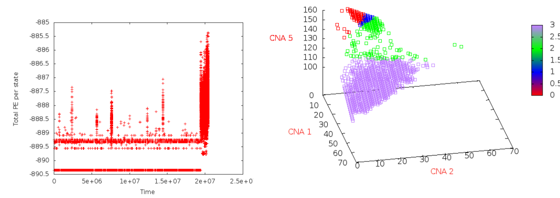
2.2.1 Particle Trajectories. The Vector Particle in Cell (VPIC) simulation studies particle accelerations within magnetic fields in plasma physics[1]. Trajectory analysis of position(where) and time(when) of the most energetic particles can help scientists to understand what causes the acceleration and use this information for further applications. To get trajectories of the top m energetic particles, we used extra *Sort* by energy on the last time step to get *particle_ids*. Then we used *Filter* to all time steps to extract those particles of interests. The trajectory plots are omitted here because of the space limits.

3 EXPERIMENTS AND RESULTS

We conducted the following experiments on the *scaling* partition of the Darwin cluster at LANL. Each node has 36 cores of Intel Broadwell E5-2695 CPU with 2 hyperthreads and 125 GB memory. The cluster is connected by InfinityBand EDR.

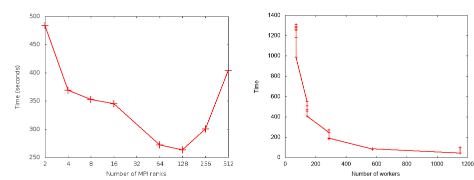
We first verified the **correctness** of our Thrill implementation. Figure 2(a) and figure 2(b) illustrate the time series and K-means clustering respectively of the AMD states. Figure 2(a) shows the time series of states and their potential energies describe system transitions and the shape of potential energy surfaces. Figure 2(b) shows clustering that groups similar states based on their CNA characteristic. From the VPIC simulation data, we got the position and time trajectory of the particle with the highest energy. Due to the space limits, we present the trajectory graphs in the supplementary materials and the poster. All the results were verified by the scientists at LANL.

Next, we verified the **scalability** of the Thrill implementation. For AMD simulations, we tested the scalability of time series and clustering analysis from 60,000 states up to 512 workers. For VPIC simulations, we tested the scalability from one single 75GB HDF5 particle file that contains 2011 time steps of the movement of 1258496 particles in each step up to 1152 workers. Figure 3 illustrates that the Thrill implementation has some strong scalability for both applications.



(a) time series (b) K-means clustering operation.

Figure 2: (a)total potential energy show an interesting transition starting around 2×10^7 time-units. (b)Four clusters in different colors.



(a) AMD Scalability (b) VPIC Scalability

Figure 3: (a)Scalability for MD. (b)Scalability for VPIC.

4 CONCLUSIONS

We explored the Thrill library to process and analyze simulation data and showed the usefulness of the declarative language paradigm in solving big data scientific problems. This helped scientists to analyze data with many files in a more efficient way. The approach displayed strong scalability. Our future work includes apply the framework to other scientific data analysis.

ACKNOWLEDGMENTS

The authors would like to thank our collaborators at LANL: John Woodring (CCS division), Danny Perez, Fan Guo and Xiaocan Li (T division); as well as Data Science at Scale School and ISTI for funding.

REFERENCES

[1] Fan Guo et al. 2015. Particle acceleration and plasma dynamics during magnetic reconnection in the magnetically dominated regime. *The Astrophysical Journal* 806, 2 (2015), 167.
[2] Rao Huang et al. 2017. Cluster analysis of accelerated molecular dynamics simulations: A case study of the decahedron to icosahedron transition in Pt nanoparticles. *The Journal of chemical physics* 147, 15 (2017), 152717.
[3] Timo Bingmann et al. 2016. Thrill: High-performance algorithmic distributed batch data processing with c++. In *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 172–183.
[4] Tao Gao, Yanfei Guo, Boyu Zhang, Pietro Cicotti, Yutong Lu, Pavan Balaji, and Michela Taufer. 2017. Mimir: Memory-efficient and scalable mapreduce for large supercomputing systems. In *Parallel and Distributed Processing Symposium (IPDPS), 2017 IEEE International*. IEEE, 1098–1108.
[5] Zhao Zhang, Kyle Barbary, Frank Austin Nothaft, Evan Sparks, Oliver Zahn, Michael J Franklin, David A Patterson, and Saul Perlmutter. 2015. Scientific computing meets big data technology: An astronomy use case. In *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE, 918–927.