Big Data Machine Learning and Graph Analytics: Current State and Future Challenges

H. Howie Huang Hang Liu
Department of Electrical and Computer Engineering
George Washington University
Email: {howie, asherliu}@gwu.edu

Abstract—**Big data machine learning and graph analytics have been widely used in industry, academia and government. Continuous advance in this area is critical to business success, scientific discovery, as well as cybersecurity. In this paper, we present some current projects and propose that next-generation computing systems for big data machine learning and graph analytics need innovative designs in both hardware and software that provide a good match between big data algorithms and the underlying computing and storage resources.**

Keywords: Big Data; Lambda Architecture; Hardware and Software Co-Design; Graphics Processing Unit; Non-Volatile Memory; Solid-State Drive

Big data computing, already a market of seven billion dollars in 2011, is projected to increase to 50 billion dollars within six years [1]. It is crucial to the success of not only internet companies, e.g. Amazon, Twitter and Facebook, but also traditional business such as Walmart and Bank of America, as well as government agencies. Furthermore, big data computing has become such a powerful paradigm that enables scientists across different disciplines to tackle challenging research problems. Two of most important big data applications are machine learning and graph analytics. For example, machine learning algorithms, e.g., collaborative filtering and topic modeling, are often used to improve user experience and increase the revenue [2], [3], [4], [5]. In the meantime, graph algorithms, such as Breath-First Search (BFS) and betweenness centrality, can be utilized for social network analysis and computational biology [6], [7], [8], [9], [10], [11], [12], [13].

Current big data computing systems fall into two major categories: *batch processing* (e.g., MapReduce and GraphLab) is able to analyze large volumes of on-disk data, but the processing time can be as long as several days and weeks; and *streaming processing* (e.g., Storm) can analyze in-memory data in a short period to time like milliseconds [14]. While batch processing focuses on the large amount of historical data (*Volume*), streaming processing deals with the instantly generated data streams (*Velocity*). Both also need to address the issues like different data types (*Variety*) and uncertainty (*Veracity*) [15], [16], [17], [18].

Recently the Lambda Architecture shown in Figure 1 is proposed to combine the capability of batch and streaming processing for next-generation big data computing systems [19]. The insight (the result of big data processing) is generated by merging the results from both pipelines. The lambda architecture, albeit an innovative design in itself, needs to tackle multiple challenges as big data continue to grow at an unexpected speed.

First, one needs to efficiently merge the models constructed from batch and streaming processing. The merging method may be vastly different for various algorithms. For example, WordCount only requires adding the values of the same key together from batch and streaming processing. However, for BFS, the newly added edges may lead to drastic changes in the traversal paths. New interfaces shall be developed to provide good flexibility and usability for application programmers.

Second, as multi-core CPUs become pervasive, hardware computational accelerators are promising in providing additional boost to the overall system performance. In recent years, a number of notable projects [20], [21], [22], [23], [24], [25], [26], including ours [27], [28], [29], have successfully utilized Graphics Processing Unit (GPU) and Many Integrated Core (MIC) architecture in different application domains. Current implementations of machine learning and graph analytics algorithms are mostly developed to run on multicore CPUs. Our research among others has shown that hardware accelerators like GPUs can provide substantial speedup over CPU for both computation and memory intensive applications. We believe that machine learning and graph algorithms are good candidates for GPU and MIC processing, and can potentially achieve a variety of benefits such as faster response time and better energy efficiency (which is another key system design issue).

Third, high-performance storage systems are needed to store and manage both in-memory and on-disk data. The availability of non-volatile memory (NVM) technology such as Flash memory, Solid-State Drive (SSD), and Phase Change Memory (PCM) presents an exciting opportunity for optimizing I/O performance and improving data processing speed. Built upon our prior work [30], [31], [32], we are in the process of designing and developing new memory and storage architectures that can store large in-memory datasets and deliver short I/O latency, while ensuring high reliability.

To summarize, next-generation computing systems for big data machine learning and graph analytics shall take full advantage of hardware accelerators and non-volatile memory, and deliver high-performance computing and storage services to big data applications.

978-1-4799-5666-1/14/$31.00 ©2014 IEEE

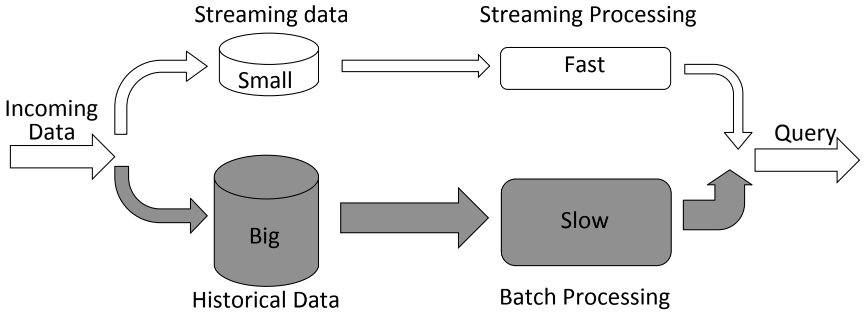

Fig. 1. Overview of the Lambda Architecture

ACKNOWLEDGMENT

This work is supported in part by National Science Foundation grants 1350766, 1124813, and 0937875.

REFERENCES

[1] Jeff Kelly. Big data vendor revenue and market forecast. Wikibon, 2014.
[2] Yunhong Zhou, Dennis Wilkinson, Robert Schreiber, and Rong Pan. Large-scale parallel collaborative filtering for the netflix prize. In *Algorithmic Aspects in Information and Management*, pages 337–348. Springer, 2008.
[3] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research, 2003*.
[4] Amol Ghoting, Rajasekar Krishnamurthy, Edwin Pednault, Berthold Reinwald, Vikas Sindhwani, Shirish Tatikonda, Yuanyuan Tian, and Shivakumar Vaithyanathan. Systemml: Declarative machine learning on mapreduce. In *IEEE International Conference on Data Engineering (ICDE), 2011*, pages 231–242.
[5] Yucheng Low, Danny Bickson, Joseph Gonzalez, Carlos Guestrin, Aapo Kyrola, and Joseph M Hellerstein. Distributed graphlab: a framework for machine learning and data mining in the cloud. *Proceedings of the VLDB Endowment (VLDB), 2012*.
[6] Christian Doerr and Norbert Blenn. Metric convergence in social network sampling. In *Proceedings of the 5th ACM workshop on HotPlanet, 2013*, pages 45–50.
[7] G Chin, Grant C Nakamura, Daniel G Chavarria, and Heidi J Sofia. Graph mining of networks from genome biology. In *Proceedings of the 7th IEEE International Conference on Bioinformatics and Bioengineering (BIBE) 2007.*, pages 1265–1269.
[8] Grzegorz Malewicz, Matthew H Austern, Aart JC Bik, James C Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. Pregel: a system for large-scale graph processing. In *Proceedings of the ACM SIGMOD International Conference on Management of data (SIGMOD), 2010*, pages 135–146.
[9] Joseph E Gonzalez, Yucheng Low, Haijie Gu, Danny Bickson, and Carlos Guestrin. Powergraph: Distributed graph-parallel computation on natural graphs. In *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2012*, volume 12, page 2.
[10] Aapo Kyrola, Guy E Blelloch, and Carlos Guestrin. Graphchi: Large-scale graph computation on just a pc. In *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2012*, volume 12, pages 31–46.
[11] Donald Nguyen, Andrew Lenharth, and Keshav Pingali. A lightweight infrastructure for graph analytics. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP), 2013*, pages 456–471.
[12] Amitabha Roy, Ivo Mihailovic, and Willy Zwaenepoel. X-stream: edge-centric graph processing using streaming partitions. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP), 2013*, pages 472–488.
[13] Zhao Zhao, Guanying Wang, Ali Raza Butt, Maleq Khan, VS Anil Kumar, and Madhav V Marathe. Sahad: Subgraph analysis in massive networks using hadoop. In *IEEE International Parallel & Distributed Processing Symposium (IPDPS), 2012*.
[14] Nathan Marz. Storm-distributed and fault-tolerant realtime computation. https://storm.incubator.apache.org/, 2013.
[15] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *IEEE Symposium on Mass Storage Systems and Technologies (MSST), 2010*, pages 1–10.
[16] David Corrigan. Integrating and governing big data. IBM Whitepaper, 2013.
[17] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM, 2008*, 51(1):107–113.
[18] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J Franklin, Scott Shenker, and Ion Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the USENIX conference on Networked Systems Design and Implementation (NSDI), 2012*, pages 2–2.
[19] Nathan Marz. Lambda architecture. http://nathanmarz.com/blog/how-to-beat-the-cap-theorem.html, 2013.
[20] Mark Silberstein, Bryan Ford, Idit Keidar, and Emmett Witchel. Gpufs: integrating a file system with gpus. In *ACM SIGARCH Computer Architecture News*, volume 41, pages 485–498, 2013.
[21] Pramod Bhatotia, Rodrigo Rodrigues, and Akshat Verma. Shredder: Gpu-accelerated incremental storage and computation. In *Proceedings of the conference on File and Storage Technologies (FAST), 2012*, page 14.
[22] Weibin Sun, Robert Ricci, and Matthew L Curry. Gpustore: harnessing gpu computing for storage systems in the os kernel. In *Proceedings of the ACM Annual International Systems and Storage Conference (SYSTOR), 2012*, page 9.
[23] Sungpack Hong, Tayo Oguntebi, and Kunle Olukotun. Efficient parallel graph exploration on multi-core cpu and gpu. In *IEEE International Conference on Parallel Architectures and Compilation Techniques (PACT), 2011*, pages 78–88.
[24] Duane Merrill, Michael Garland, and Andrew Grimshaw. Scalable gpu graph traversal. In *ACM SIGPLAN Notices, 2012*.
[25] Rajat Raina, Anand Madhavan, and Andrew Y Ng. Large-scale deep unsupervised learning using graphics processors. In *Proceedings of the ACM International Conference on Machine Learning (ICML), 2009*, volume 9, pages 873–880.
[26] Bryan Catanzaro, Narayanan Sundaram, and Kurt Keutzer. Fast support vector machine training and classification on graphics processors. In *Proceedings of the ACM International Conference on Machine Learning (ICML), 2008*, pages 104–111.
[27] Hang Liu, Jung Hee Seo, Rajat Mittal, and H Howie Huang. Matrix decomposition based conjugate gradient solver for poisson equation. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis (SC), 2012*, pages 1499–1500.
[28] Hang Liu, Jung-Hee Seo, Rajat Mittal, and H Howie Huang. Gpu-accelerated scalable solver for banded linear systems. In *IEEE International Conference on Cluster Computing (CLUSTER), 2013*, pages 1–8.
[29] Wei Wang, Lifan Xu, John Cavazos, H Howie Huang, and Matthew Kay. Fast acceleration of 2d wave propagation simulations using modern computational accelerators. *Journal of PloS one, 2014*.
[30] H Howie Huang, Shan Li, Alex Szalay, and Andreas Terzis. Performance modeling and analysis of flash-based storage devices. In *IEEE Symposium on Mass Storage Systems and Technologies (MSST), 2011*, pages 1–11.
[31] Jie Chen, Guru Venkataramani, and H Howie Huang. Repram: Recycling pram faulty blocks for extended lifetime. In *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2012*.
[32] Ahsen J Uppal, Ron C Chiang, and H Howie Huang. Flashy prefetching for high-performance flash drives. In *IEEE Symposium on Mass Storage Systems and Technologies (MSST), 2012*.