Big Graph Analytics Systems

Da Yan
The Chinese University of Hong Kong
yanda@cse.cuhk.edu.hk

Yingyi Bu
Couchbase, Inc.
yingyi@couchbase.com

Yuanyuan Tian
IBM Almaden Research Center, USA
ytian@us.ibm.com

Amol Deshpande
University of Maryland
amol@cs.umd.edu

James Cheng
The Chinese University of Hong Kong
jcheng@cse.cuhk.edu.hk

ABSTRACT

In recent years we have witnessed a surging interest in developing Big Graph processing systems. To date, tens of Big Graph systems have been proposed. This tutorial provides a timely and comprehensive review of existing Big Graph systems, and summarizes their pros and cons from various perspectives. We start from the existing vertex-centric systems, which which a programmer thinks intuitively like a vertex when developing parallel graph algorithms. We then introduce systems that adopt other computation paradigms and execution settings. The topics covered in this tutorial include programming models and algorithm design, computation models, communication mechanisms, out-of-core support, fault tolerance, dynamic graph support, and so on. We also highlight future research opportunities on Big Graph analytics.

CCS Concepts

•Mathematics of computing → Graph algorithms; •Networks → Cloud computing; •Theory of computation → Graph algorithms analysis; •Computing methodologies → Parallel computing methodologies; •Computer systems organization → Parallel architectures;

Keywords

Graph; system; platform; Pregel; GraphLab; vertex-centric

1. INTRODUCTION

With the increasing popularity of online social networks and the prevalence of mobile communication networks and other types of information networks, there has been a great interest in developing efficient and scalable techniques to study these networks, modeled as graphs. In particular, many Big Graph systems have been developed in recent years, which provide a user-friendly, unified framework for programmers to implement parallel algorithms for all kinds of graph analytics. As graph data continue to proliferate in many real world applications, more Big Graph systems are expected to emerge in the near future. However, the sudden emergence of so many Big Graph systems easily overwhelms both researchers and general users who are interested in developing new systems or in using these systems for graph analytics.

This tutorial provides a comprehensive survey of the existing Big Graph systems. For each work (or system), we highlight the key ideas and features of its design, and summarize its strengths and weaknesses. The goal of this tutorial is three-fold. With an in-depth analysis of the various techniques adopted by existing systems, we aim to (1) help system researchers avoid reinventing the wheel, but instead, develop new ideas in system design; (2) help system practitioners apply useful existing techniques to their systems to improve system performance; and (3) help general users understand the pros and cons of different systems and select the right system for their particular graph analytics tasks.

While the existing Big Graph systems can be categorized in various ways, this tutorial categorizes these systems mainly according to their *programming models*, *execution models*, and *execution settings*. We emphasize them dimensions since user-friendliness in programming is one important feature of modern Big Graph systems, while the execution model and settings are critical to the efficiency of graph computation.

Features of Big Graph Systems. We first consider the computation model. Vertex-centric computation is probably the most popular computation model for iterative graph computations, where a programmer only needs to specify the behavior for one generic vertex, and the systems automatically apply the user-defined logic to different vertices in a graph to complete the computation. There also exist graph-parallel systems that adopt other computation models. For example, Giraph++ [9] and Blogel [12] extend the vertex-centric computation model with a novel block-centric computation model, to avoid incurring communication for the computation within a subgraph block. GRACE [11] applies a similar idea to improve the hit rate of CPU caches. Among others, PEGASUS [5] and GBASE [4] model graph computation by (generalized) matrix-vector multiplications; while Green-Marl [3] and Galois [6] let users specify the computation logic by a domain-specific language (DSL). N-Scale [7] adopts a subgraph-centric model where users define

DOI: 10.1145/1235

computation logic on a subgraph of interest, and the model is better at solving graph problems whose outputs are a set of possibly overlapping subgraphs, such as graph matching and motif mining. There are also systems that use Datalog for Big Graph analytics, such as SociaLite [8].

Other features of big data systems include (1) whether the execution is synchronous or asynchronous, (2) the support for topology mutation, i.e., the deletion and addition of edges and vertices; (3) the support for dynamic graphs with frequent updates, such as in Kineograph [1], TIDE [10], and Chronos [2]; and (4) the support for out-of-core execution. Finally, in terms of hardware platforms, while most graph systems are designed to run on a cluster, there are a few systems designed to run in a single machine or to run with SSDs or GPUs.

2. TARGET AUDIENCE

The target audience for this tutorial includes all who are interested in graph analytics and its applications.

3. TUTORIAL OUTLINE

The preferred duration of this tutorial is 3 hours, which covers two parts that consist of 6 topics (see below). Each part takes around 1.5 hours. Part I covers Topics 1 and 2, and Part II covers Topics 3, 4, 5 and 6.

3.1 Topic 1: Pregel-Like Systems

In this topic, we first review the computation model of Pregel, and introduce how to design Pregel algorithms that have performance guarantees. We then review existing Pregel-like systems that optimizes the basic model of Pregel from various aspects, including communication mechanisms, load balancing, computation model, support for asynchronous execution, and fault tolerance.

3.2 Topic 2: Shared Memory Abstraction

In this topic, we review vertex-centric Big Graph systems that adopt the shared memory abstraction. We first review the pioneering system GraphLab, as well as its improved version, PowerGraph. We then introduce three important systems that adopt shared memory abstraction: GraphChi, X-Stream and VENUS. These systems are designed to perform out-of-core graph processing on a single PC.

3.3 Topic 3: Matrix-Based Systems

In this topic, we first review three matrix-based systems that rely on MapReduce for execution: PEGASUS, GBASE and SystemML. We then review three single-machine systems that adopt a vertex-centric programming model but perform matrix-based execution: GraphMat, GraphTwist and GridGraph.

3.4 Topic 4: Single-Machine Systems

While Topics 2 and 3 already covered some single-machine systems, this topic review more single-machine systems that can be classified into three categories: (1) out-of-core systems that process graphs on SSD(s); (2) in-memory systems for multi-core execution, which emphasize more on ease of programming and high parallelism; (3) systems for execution with GPU(s).

3.5 Topic 5: Subgraph-Based Systems

This topic reviews the two recent systems that adopt a *overlapping subgraph-centric* model, NScale and Arabesque. These systems target graph analysis tasks whose output size can be exponential in the graph size (e.g., graph matching and finding motifs), or tasks which require reasoning about neighborhoods or connected components, neither of which can be well-solved by vertex-centric systems. We also point out the inefficiencies of these systems in constructing subgraphs for processing, and identify the opportunities for further research in this field.

3.6 Topic 6: DBMS-Based Systems

In this topic, we discuss connections between DBMS-style recursive query processing and Big Graph analytical systems. We revisit recent research projects that are "resurging" Datalog in the context of scalable Big Graph analytics, discuss existing dataflow-based graph systems, and dataflow systems for incremental graph processing.

4. OPEN PROBLEMS

We end our tutorial with a discussion of open problems in developing Big Graph analytics platforms. Despite the plurality of graph systems, very few support the analysis of temporal graphs that dynamically change over time. More work can be done in this area, especially for efficiently supporting dynamic graphs with frequent vertex and edge deletions. Also, existing systems for graph matching and motif finding all suffer from various performance problems, thus a more efficient framework for the overlapping subgraph-centric model remains to be found. Another interesting open problem is to study how the various models can be integrated together to solve complicated real-life problems that involve graph ETL (extract, transform, and load) and computation tasks of all kinds.

Acknowledgments. Da Yan and James Cheng are supported by the Hong Kong GRF 2150851. Amol Deshpande is supported by NSF under grant IIS-1319432, and an IBM Collaborative Research Award.

5. REFERENCES

[1] R. Cheng, J. Hong, A. Kyrola, Y. Miao, X. Weng, M. Wu, F. Yang, L. Zhou, F. Zhao, and E. Chen. Kineograph: taking the pulse of a fast-changing and connected world. In *EuroSys*, pages 85–98, 2012.

[2] W. Han, Y. Miao, K. Li, M. Wu, F. Yang, L. Zhou, V. Prabhakaran, W. Chen, and E. Chen. Chronos: a graph engine for temporal graph analysis. In *EuroSys*, pages 1:1–1:14, 2014.

[3] S. Hong, H. Chafi, E. Sedlar, and K. Olukotun. Green-marl: a DSL for easy and efficient graph analysis. In *ASPLOS*, pages 349–362, 2012.

[4] U. Kang, H. Tong, J. Sun, C. Lin, and C. Faloutsos. GBASE: a scalable and general graph management system. In *SIGKDD*, pages 1091–1099, 2011.

[5] U. Kang, C. E. Tsourakakis, and C. Faloutsos. PEGASUS: A peta-scale graph mining system. In *ICDM*, pages 229–238, 2009.

[6] D. Nguyen, A. Lenharth, and K. Pingali. A lightweight infrastructure for graph analytics. In *SOSP*, pages 456–471, 2013.

[7] A. Quamar, A. Deshpande, and J. Lin. Nscale: Neighborhood-centric analytics on large graphs. *PVLDB*, 7(13):1673–1676, 2014.

[8] J. Seo, S. Guo, and M. S. Lam. Socialite: Datalog extensions for efficient social network analysis. In *29th*

IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013, pages 278–289, 2013.

[9] Y. Tian, A. Balmin, S. A. Corsten, S. Tatikonda, and J. McPherson. From "think like a vertex" to "think like a graph". *PVLDB*, 7(3):193–204, 2013.

[10] W. Xie, Y. Tian, Y. Sismanis, A. Balmin, and P. J. Haas. Dynamic interaction graphs with probabilistic edge decay. In *ICDE*, pages 1143–1154, 2015.

[11] W. Xie, G. Wang, D. Bindel, A. J. Demers, and J. Gehrke. Fast iterative graph computation with block updates. *PVLDB*, 6(14):2014–2025, 2013.

[12] D. Yan, J. Cheng, Y. Lu, and W. Ng. Blogel: A block-centric framework for distributed computation on real-world graphs. *PVLDB*, 7(14):1981–1992, 2014.

6. BIOGRAPHIES

Dr. YAN, Da is with the Department of Computer Science and Engineering at the Chinese University of Hong Kong. He is the winner of the 2015 Hong Kong Young Scientist Award.

Dr. Yingyi Bu is a senior software engineer in Couchbase Inc. He is the recipient of a Google PhD fellowship award and a Yahoo! key scientific challenge award.

Dr. Yuanyuan Tian is a Research Staff Member at IBM Almaden Research Center, USA. Her research interests include large scale machine learning, graph analytics, SQL on Hadoop, and big data federation. She is the recipient of the Distinguished Achievement Award from the Univeristy of Michigan.

Dr. Amol Deshpande is an Associate Professor of Computer Science at the University of Maryland at College Park. His research interests include graph data management, adaptive query processing, data streams, sensor networks and statistical modeling of data. He is a recipient of the NSF CAREER Award.

Dr. James Cheng is with the Department of Computer Science and Engineering at the Chinese University of Hong Kong. His research focuses on big data infrastructures, distributed computing systems, and large-scale network analysis.