Graph Synopses, Sketches, and Streams: A Survey

Sudipto Guha
University of Pennsylvania
3330 Walnut Street
Philadelphia, PA, USA
sudipto@seas.upenn.edu

Andrew McGregor
University of Massachusetts Amherst
140 Governor's Drive
Amherst, MA, USA
mcgregor@cs.umass.edu

ABSTRACT

Massive graphs arise in any application where there is data about both basic entities and the relationships between these entities, e.g., web-pages and hyperlinks; neurons and synapses; papers and citations; IP addresses and network flows; people and their friendships. Graphs have also become the de facto standard for representing many types of highly structured data. However, the sheer size of many of these graphs renders classical algorithms inapplicable when it comes to analyzing such graphs. In addition, these existing algorithms are typically ill-suited to processing distributed or stream data.

Various platforms have been developed for processing large data sets. At the same time, there is the need to develop new algorithmic ideas and paradigms. In the case of graph processing, a lot of recent work has focused on understanding the important algorithmic issues. An central aspect of this is the question of how to construct and leverage small-space synopses in graph processing. The goal of this tutorial is to survey recent work on this question and highlight interesting directions for future research.

1. INTRODUCTION

Consider telephone call networks: the number of calls in a single day is around 10^9 and this information is stored for billing purposes. Processing even a few days' worth of billing data requires specialized techniques. The Web graph has about 10^{10} nodes which are rich in attributes. Network traffic data averages to about 10^9 packets per hour per router for large ISPs; storing all this information is a poor use of resources and processing the data offline is often unacceptable from the perspective of latency. Large scale scientific experiments generate terabytes of data in a span of days. At these scales, even constructing and storing a traditional representation of a graph can be a significant challenge; performing heavy-duty computation and evaluating complex queries requires radically new approaches.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Articles from this volume were invited to present their results at The 38th International Conference on Very Large Data Bases, August 27th - 31st 2012, Istanbul, Turkey.
Proceedings of the VLDB Endowment, Vol. 5, No. 12
Copyright 2012 VLDB Endowment 2150-8097/12/08... $ 10.00.

While a number of promising platforms have been introduced to manage large scale graph processing (see, e.g., Giraph [4], GraphLab [5] and Pregel [6]), there is also the need to develop new algorithmic ideas and paradigms for these platforms.

One useful model for developing algorithms for massive graphs is the *semi-streaming model* [3, 7]. This model has been studied for nearly ten years and there has been a burst of recent progress. Through the study of this model, a rich set of techniques have been developed that capitalize on, and contribute to, research in a wide variety of other areas such as compressed sensing, property testing and learning, distributed processing and MapReduce, sampling and statistical estimation. As such, the guiding philosophy of this tutorial idea is not to focus on a specific model (e.g., single-pass stream processing) but rather on abstracting and developing the principles of compressed representations, or data synopses, and working with them.

Over the last couple of decades, there have been big strides in the area of streaming algorithms for processing numeric data [7] and XML data [8]. However, the task of processing general graphs, which poses some of the richest questions, has received significantly less attention. For example, there now exist many techniques for summarizing numerical data and feature vectors including sketching, hashing, histograms, and low-dimensional approximations. What are the equivalent synopses for graph data? If they exist, how can they be constructed given a data stream, or distributed data, or samples from a data?

The goals of this tutorial are to a) survey the recent progress and b) initiate a dialogue of what next steps or directions are to be pursued. In what follows, we give a brief outline of the contents of the tutorial.

1.1 Part I: Models & Basic Graph Queries

We start with an overview of the characteristics of some common massive graphs such as the web-graph, call-graph, social networks, and IP networks. We then formally define two relevant models for processing massive graphs: the *sketching* and *streaming models*.

To illustrate the important concepts and algorithmic techniques in these models, we initially focus on the basic problem of determining whether a massive graph is connected, i.e., whether there is a path between any two nodes. Considering this foundational problem will lead us to a series of important definitions and techniques including the semi-streaming model. We then address three related problems:

1. *Shortest Paths:* How long is the shortest path between

a given pair of nodes? Can we efficiently approximate the set of all graph distances without storing the entire graph?

2. *Robust Connectivity:* How well connected are a given pair of nodes, e.g., are there many disjoint paths between the nodes or could the removal of a few edges disconnect the nodes.

3. *Dynamic Graphs:* Can we answer such questions for massive graphs defined by both insertions and deletions of edges?

In addressing each question we will need to define and investigate different forms of graph synopses. This includes *spanners* and *sparsifisers*; these are sparse (weighted) subgraphs of the original graph from which properties such as the lengths of the shortest paths and the size of graph cuts can be approximated. What is surprising is not only that such small synopses exist but also that they can be constructed efficiently in the data stream model. In presenting such constructions, we demonstrate various general techniques such as geometric grouping and reduce-and-merge trees.

The question of dealing with insertions and deletions naturally leads up to *sketch synopses*, i.e., linear projections of the data into lower dimensional spaces that preserve the salient features of the data. We will discuss the recent work that introduces the notion of *graph sketches* and its application to both processing graph streams and more widely, e.g., in MapReduce and other parallel and distributed settings [1, 2].

1.2 Part II: More Advanced Queries

In the second part of the tutorial we focus on more advanced computational problems. We base our presentation around two popular graph processing tasks.

1. *Graph Matchings:* How do we pair up adjacent nodes of a graph such that the maximum number of nodes are paired? This classical problem arises in many guises such as pairing doctors to medical residences, assigning ad slots to advertisers, and refining finite element meshes.

2. *Graph Clustering:* Given a graph suppose that the nodes represent various entities and that the weighted edges between nodes indicate the similarity between the nodes, e.g., an edge between two nodes would have a negative value if the two nodes represent entities that are dissimilar whereas a positive weight would indicate that the entities are very similar. The goal is to partition the graph such that similar nodes stay in the same partition but dissimilar nodes are separated. The problem of graph clusterings, also known an *correlation clustering*, is a natural next step in a long line of clustering problems that have been addressed the data stream model.

In the course of investigating these two classic problems we demonstrate the utility of various linear programing techniques when combined with the basic sparsifier synopsis discussed in the first part of the tutorial. Lastly, we overview some work on processing graphs in variants of the data stream model including the annotated stream model and the random order stream model. We conclude by highlighting a selection of open problems in the area of graph streams and sketches.

Acknowledgments

We would like to acknowledge NSF awards CCF-0644119, CCF-0953754, and CCF-1117216.

2. BIOGRAPHICAL SKETCHES

Sudipto Guha is an Associate Professor in the Department of Computer and Information Sciences at University of Pennsylvania since Fall 2001. He completed his Ph.D. in 2000 at Stanford University working on approximation algorithms. He is a recipient of the NSF CAREER award in 2007, and the Alfred P. Sloan Foundation fellowship.

Andrew McGregor is an Assistant Professor of Computer Science of the University of Massachusetts, Amherst. He received his Ph.D. from the University of Pennsylvania in 2007 and spent two years as a postdoctoral researcher, first at UC San Diego and then at Microsoft Research SVC. He was awarded the NSF CAREER Award in 2010 for his work on data streams and sketching.

3. REFERENCES

[1] K. J. Ahn, S. Guha, and A. McGregor. Analyzing graph structure via linear measurements. In *ACM-SIAM Symposium on Discrete Algorithms*, 2012.

[2] K. J. Ahn, S. Guha, and A. McGregor. Graph sketching: Sparsification, spanners and subgraphs. In *ACM Symposium on Principles of Database Systems*, 2012.

[3] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang. On graph problems in a semi-streaming model. *Theor. Comput. Sci.*, 348(2):207–216, 2005.

[4] Giraph. http://incubator.apache.org/giraph/.

[5] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein. GraphLab: A new parallel framework for machine learning. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, Catalina Island, California, July 2010.

[6] G. Malewicz, M. H. Austern, A. J. C. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski. Pregel: a system for large-scale graph processing. In *SIGMOD Conference*, pages 135–146, 2010.

[7] S. Muthukrishnan. *Data Streams: Algorithms and Applications*. Now Publishers, 2006.

[8] D. Olteanu. *Evaluation of XPath Queries against XML Streams*. PhD thesis, University of Munich, 2004.