Research Proposal

Realtime property evaluation of Large Streaming Graphs

O.I. Mudannayake
Index number: 15000893
Registration number: 2015CS089

Dr. D.N. Ranasinghe
(Supervisor)

University of Colombo School of Computing

Signatures of Researcher and Supervisor

.....
O.I. Mudannayake
Researcher

.....
Dr. D.N. Ranasinghe
Supervisor

Contents

List of Figures

List of Tables

1.0 Introduction

Massive scale datasets are becoming increasingly common today. Growth of the number of users who are actively using digital devices connected to the internet have vastly affected this phenomenon. Also there lies an interest in researchers to solve the problems which involves large datasets. Most of these datasets could be mapped to graphs to extract useful information, giving rise to the need of processing massive scale graphs. There are many practical scenarios where massive scale graphs are applied such as social networks, network traffic data and road networks.

It is much easier to work with graphs when they are static and small. However most of the natural graphs that are being encountered in the real world are dynamic. It becomes increasingly complex to handle the graph as the velocity with which its edges gets updated increases. Large scale dynamic natural graphs are used by many companies today. Google uses PageRank algorithm[2] to map the links between the web pages. And facebook has a massive graph depicting the connections of each user on the platform.

With the size of the massive scale graphs, it is difficult to evaluate the properties of them even after partitioning into multiple nodes. The graphs have to be summarized so that important information regarding the underlying dataset can be inferred easily.

Motivation

Being applied in a wide range of industrial and research applications, realtime property evaluation of streaming and dynamic natural graphs is a critical requirement in many scenarios. Graph summarization plays a big role in this as it reduces the computational resources required to evaluate the properties in a rather massive scale steaming graph. It would be beneficial for number of sectors in the process of summarizing streaming graphs were made efficient.

1.1 Related work

Graph partitioning

The size of modern datasets have become too large to be fit into a single machine. It has become unrealistic to try to process the graphs mapped to these massive datasets while keeping them in the memory of a single node. Hence there exist a need to partition those large scale graphs into multiple machines. However graph partitioning has been proved to be an NP-hard problem[3]. Therefore the graph partitioning algorithms are only able to give sub-optimal solutions as of this date. Graph partitioning algorithms can be mainly divided into two, such that, Online Graph Partitioning Algorithms and Offline Graph Partitioning

Algorithms. The offline partitioning algorithms such as METIS[4], Chaco[5], SBV-cut[6] need to load the entire graph into the memory for the algorithm to be run. But the online partitioning algorithms like PreferBig[7] and HoVerCut[8] keeps a buffer of the edge streams and process them when the buffer gets full.

Graph summarization

		Input Graph				Algorithmic Properties				
	Method	Weighted	Undirect.	Directed	Heterog.	Prm-free	Linear	Technique	Output	Objective
Static Plain Graphs	GraSS (LeFevre and Terzi 2010)	✘	✔	✘	✘	✘	✘	grouping	supergraph	query efficiency
	Weighted Compr. Toivonen et al. (2011)	✔	✔	✔*	✘	✘	✘	grouping	supergraph	compression
	coarseNet (Purohit et al. 2014)	✔	✘	✔	✘	✘	✔*	grouping	supergraph	influence
	l_p-reconstr. Error (Riondato et al. 2014)	✔	✔	✘	✘	✘	✔	grouping	supergraph	query efficiency
	Motifs (Dunne and Shneiderman 2013)	✘	✔	✘	✔	✘	✘	grouping	supergraph	visualization
	CoSum (Zhu et al. 2016)	✔*	✔	✘	✔	✔	✔	grouping	supergraph	entity resolution
	Dedensification (Maccioni and Abadi 2016)	✘	✔*	✔	✔	✔	✔*	(edge) grouping	sparsified graph	query efficiency
	VNM (Buehrer and Chellapilla 2008)	✘	✘	✔	✘	✘	✘	(edge) grouping	sparsified graph	patterns
	MDL Repres. (Navlakha et al. 2008)	✘	✔	✔*	✘	✔	✘	compression	supergraph	compression
	VoG (Koutra et al. 2014a)	✘	✔	✘	✘	✔	✔*	compression	structure list	patterns, visualiz.
	OntoVis (Shen et al. 2006)	✘	✔	✘	✔	✔	✔	simplification	sparsified graph	visualization
	Egocentric Abstr. (Li and Lin 2009)	✘	✘	✔	✔	✘	✘	simplification	sparsified graph	influence
	CSI (Mehmood et al. 2013)	✘	✘	✔	✘	✔	✘	influence	supergraph	influence
	SPINE (Mathioudakis et al. 2011)	✔	✘	✔	✘	✘	✘	influence	sparsified graph	influence
Static Labeled Graphs	S-Node (Raghavan and Garcia-Molina 2003)	✘	✘	✔	✘	✔	✘	grouping	supergraph	query efficiency
	SNAP/k-SNAP (Tian et al. 2008)	✘	✔	✔*	✘	✔	✔	grouping	supergraph	query efficiency
	CANAL (Zhang et al. 2010)	✘	✔	✔*	✘	✔	✘	grouping	supergraph	patterns
	Probabilistic (Hassanlou et al. 2013)	✔	✘	✔	✘	✔	✔	grouping	supergraph	compression
	Query-Pres. (Fan et al. 2012)	✘	✘	✔	✘	✘	✔	grouping	supergraph	query efficiency
	ZKP (Shoaran et al. 2013)	✘	✘	✔	✘	✔	✔	grouping	supergraph	privacy
	Randomized (Chen et al. 2009)	✘	✔	✘	✔	✔	✘	grouping	supergraph	patterns
	d-summaries (Song et al. 2016)	✘	✘	✔	✔	✘	✘	grouping	supergraph	query efficiency
	SUBDUE (Cook and Holder 1994)	✘	✔	✔	✔	✔	✘	compression	supergraph	patterns
	AGSUMMARY (Wu et al. 2014)	✘	✘	✔	✘	✔	✔	compression	supergraph	compression
	LSH-based (Khan et al. 2014)	✘	✘	✔	✘	✘	✔	compression	supergraph	compression
	VEGAS (Shi et al. 2015)	✔*	✘	✔	✘	✘	✔*	influence	supergraph	influence
Dynamic Graphs	NetCondense(Adhikari et al. 2017)	✔	✔	✔	✘	✘	✘	grouping	temporal supergraph	influence
	TCM (Tang et al. 2016)	✔	✔	✔	✘	✘	✔	grouping	supergraph	query efficiency
	TimeCrunch (Shah et al. 2015)	✘	✔	✘	✔	✘	✔	compression	ranked list of temporal structures	temporal patterns, visualization
	OSNet (Qu et al. 2014)	✘	✔	✘	✘	✘	✔	influence	subgraphs of diffusion over time	influence
	Social Activity (Lin et al. 2008)	✘	✔	✘	✔	✘	✘	influence	temporal themes	influence, visualization

Figure 1: Qualitative Comparison of All Graph Summarization Techniques Based on the Properties of the Input Graph[1]

There exist different techniques to summarize various types of graphs, i.e, static, dynamic, heterogeneous, streaming and domain dependant graphs. Summarizing each of these graphs

prove to be a unique challenge while the objectives of summarization vary as well. Graph summarization is always a tradeoff between accuracy, efficiency and space.

Some methods have already been devised to summarize the graph streams such as gSketch[9], GMatrix[10] and TCM[11]. These techniques permit only some types of queries to be run against the summaries with varying degrees of accuracies. This research aims to further analyze the properties of streaming graphs through summarization.

2.0 Research Questions

Restating the aforementioned, there already exists some techniques of summarizing streaming graphs such as gSketch, GMatrix and TCM. And the efficiency of these methods have been also mentioned in the referenced texts with respect to various queries, i.e, edge queries, node queries and path queries. Despite the improvements of the recent summarization methods like TCM, all these techniques pose tradeoffs with regard to evaluating various graph properties and only a small portion of them have been tested in the original research work. Therefore the performance of these summarization techniques must be compared against each other while evaluating other graph properties such as KNN and K-furthest neighbor. Much research has been remaining on optimizing the streaming graph summarization techniques with respect to each of those graph properties.

In the original work proposed through "An efficient query platform for streaming and dynamic natural graphs"[12], which was the initial motivation for this research, states future work as, "The automatic sketch creation mechanism should be improved with identifying better heuristics for deciding when to created a new sketch. And also should identify if any better mechanisms for creating and updating new sketches exists", hinting that much work has to be done with regard to automatic sketch creation in streaming graph summarization. Furthermore it adds that "This is an embarrassingly parallel query framework model, therefore the model should be evaluated on top of a parallel framework. And also, new metrics has to be defined for thorough evaluation". This suggests that the proposed solution has yet to be implemented parallely in a suitable environment and be evaluated.

To summarize the research questions,

- How to optimize the streaming graph sketching process such that graph properties could be identified in realtime?
- How to modify the previous work[12] such that it runs on a parallel query framework?

3.0 Aims

- Discover and improve upon graph summarization techniques to derive properties of streaming graphs in realtime.

4.0 Objectives

1. Evaluate the performance of existing streaming graph summarization techniques with respect to different graph properties.
2. Build upon the work of existing sketching methods and propose improvements on creating and updating sketches.
3. Evaluate the query performance of summarized graphs generated by improved sketching techniques in deriving graph properties.
4. Extend the work done by "An efficient query platform for streaming and dynamic natural graphs"[12], to be evaluated on a parallel query framework.

5.0 Contribution to Computer Science and the Society

The research, "Real time property evaluation of Large Streaming Graphs" touches many areas in Computer Science such as Graphs, Distributed Computing, Parallel Computing.

There exists many sketching techniques for large streaming and dynamic graphs. The research will mainly improve those existing techniques and produce results comparing the improvements with the existing solutions. During the research, we will also try to explore the possibility of introducing a new sketching methods which gives better performance than the existing methods.

This project adds value to the scientific community by extending the original work "An efficient query platform for streaming and dynamic natural graphs"[12]. There is much future work left in the original research. And we attempt to develop solutions for a portion of the future work stated in the aforementioned research.

Massive scale streaming graphs are widely used in the industry today. Devising more efficient ways to evaluate their properties by applying sketching techniques will benefit many people. Also it will save the computational resources spent on evaluating these massive streaming graphs by a significant margin.

6.0 Methodology

The project will deal with large scale streaming and dynamic graphs. Therefore the first step would be obtaining available dataset of large natural graphs. One type of dataset that we will mainly focus on is the interactions of users in a social network. There are existing crawlers which are able to scrape things like tweets and facebook posts. We will also use a web crawler as suggested in the original work, to build a graph while the crawler is traversing through the web. "This is an unbounded graph since we do not know how large the graph would be and the graph keeps building while the crawler keeps traversing."[11] And when it is within the acceptable boundary conditions of the research, we will use synthetic graphs to test out the algorithms and various corner cases.

Then we will spend time on preparing an adequate environment for the sketching and partitioning operations to be run. There exists number of graph frameworks to facilitate graph computations such as Pregel, GraphLab, Distributed GraphLab, Giraph, Giraph++. And data streaming frameworks like Apache Flink will be studied and used for the development purposes.

Many streaming graph partitioning and sketching algorithms proposed by the previous researches haven't made the codebase available along with the paper. So these will be re-implemented to be run on our environment of choice. In the researches where the sketching algorithms were proposed, there were evaluated against different types of queries, i.e edge queries, node queries and path queries. Reevaluating those with a similar set of parameters will help us in clearly differentiating the strengths and weaknesses of each sketching method.

Hereafter, the implemented sketching mechanisms will be tested in evaluating other graph properties like KNN and K-furthest neighbor. There are various evaluating techniques proposed by the previous researches in evaluating the sketching mechanisms such as Average Relative Error and Number of Effective Queries[12]. And other resource factors such as memory usage will also be measured in each test.

After proper evaluation of the strengths and weaknesses of existing methods with respect to different graph properties, we will research on improving those methods. And we will attempt at proposing how which technique performs better when evaluating each graph property.

In the final phase of the project, we will focus on evaluating the models on top of a parallel framework as the future work of the original research suggests[12]. And then we will re-evaluate the proposed improvements on top of the parallel framework to report any performance gained in running them parallelly.

7.0 Project Scope and Delimitations

7.1 In Scope

1. Project will improve upon the existing streaming graph sketching techniques.
2. Efficiency of running queries on various graph properties after the sketching process will be improved.
3. New sketching techniques will be evaluated on a parallel framework.

7.2 Out of Scope

There exists number of methods for graph partitioning. The effect of these partitioning techniques will affect the results of the query efficiency and thus they will be considered during the experimentation phase and in presenting results. However no significant work or separate evaluation will be done with regard to graph partitioning. Improving partitioning techniques will be considered as out of scope for this research.

Graph summarization could be mainly subdivided into Aggregation based, Attribute based, Compression and Application oriented. This research won't explore the compression type graph summarization techniques. Much focus will be given to aggregation based sketching techniques.

There are many security concerns with regard to the partitioning of streaming graphs and running sketching algorithms in a distributed environment. However the security aspects of sketch creation and partitioning will be considered as out of scope for this research as they are trivial to the holistic view of the research question, i.e, efficient sketching of streaming graphs.

To summarize, following work will be considered as out of scope for this project.

1. Graph partitioning methods will not be improved upon.
2. Security aspects of the above algorithms with regards to a distributed environment will not be considered.

8.0 Project Timeline

M	Milestone	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov
M1	Research topic selection	x									
	Literature review	x	x	x	x	x	x	x	x	x	
	Draft the project proposal		x								
M2	Preparing the test environment and the datasets			x							
	Implement the sketching algorithms			x							
	Initial evaluation				x						
M3	Propose improvements					x	x	x			
	Mid evaluation					x	x	x			
M4	Run on a parallel framework								x		
	Final evaluation								x		
M5	Writing the project thesis									x	x
	Publication of the research									x	x

Table 1: Tentative timeline

9.0 Budget

Cloud computing resources would have to be purchased depending on the demand, in order to run the proposed algorithms on large streaming graphs.

Below is the pricing of VM instances on AWS at the time of writing the research proposal[13].

	Amazon EC2 Price	Amazon EMR Price
General Purpose - Current Generation		
m5.xlarge	$0.192 per Hour	$0.048 per Hour
m5.2xlarge	$0.384 per Hour	$0.096 per Hour
m5.4xlarge	$0.768 per Hour	$0.192 per Hour
m5.12xlarge	$2.304 per Hour	$0.27 per Hour
m5.24xlarge	$4.608 per Hour	$0.27 per Hour
m5a.xlarge	$0.172 per Hour	$0.043 per Hour
m5a.2xlarge	$0.344 per Hour	$0.086 per Hour
m5a.4xlarge	$0.688 per Hour	$0.172 per Hour
m5a.12xlarge	$2.064 per Hour	$0.27 per Hour
m5a.24xlarge	$4.128 per Hour	$0.27 per Hour
m5d.xlarge	$0.226 per Hour	$0.057 per Hour
m5d.2xlarge	$0.452 per Hour	$0.113 per Hour
m5d.4xlarge	$0.904 per Hour	$0.226 per Hour
m5d.12xlarge	$2.712 per Hour	$0.27 per Hour
m5d.24xlarge	$5.424 per Hour	$0.27 per Hour
m4.large	$0.10 per Hour	$0.03 per Hour
m4.xlarge	$0.20 per Hour	$0.06 per Hour
m4.2xlarge	$0.40 per Hour	$0.12 per Hour
m4.4xlarge	$0.80 per Hour	$0.24 per Hour
m4.10xlarge	$2.00 per Hour	$0.27 per Hour
m4.16xlarge	$3.20 per Hour	$0.27 per Hour

Figure 2: Cost of AWS VM instances

10.0 Personal Statement

This project is involved with Graphs, which is one of my main areas of interests in Computer Science. Also it is an optimization problem, which I highly enjoy working on. And the fact that the project involves with parallel computing, large data streams made it an obvious choice for me to choose it as the 4th year research project. In apart from all the CS aspects regarding the project, what motivated me to choose the problem was that it addresses an increasing issue in the modern world related to massive datasets. This will benefit lot of industrial as well as research applications which deals with large amounts of data. If I were able to make even a small change that would positively addresses such an issue, that would fulfill my objective in undertaking this project.

11.0 Research standards & tools for quality research

This sections includes my awareness and usage of academic writing, reference managing software, referencing formats, plagiarism detectors as per the guidelines of the course ***SCS4124 - Final Year Project in Computer Science***.

11.1 Academic writing

I haven't done much academic writing before except for writing of a technical article regarding ***Machine Learning over Encrypted Data***. However I have thoroughly read the best practices that should be followed when engaged in academic writing. I was able to learn much about the academic writing styles by reading lot of research papers and white papers.

11.2 Reference managing software

Mendeley

This was my initial choice for reference management. However the 1 GB cloud sync limit posed a big problem to me as I wanted to save lot of research materials. Mendeley was a good software, but I identified some bugs when I tried to update the software.

Zotero

I was introduced to Zotero by one of my colleagues. And it initially attracted me because of the rich set of options that it had to manage a hard copy of the research articles. The software allows users to store either a soft link to the file or a hard copy. So this allowed me to keep a backup of my Zotero library and sync the folder to my own cloud storage which had much more capacity. Also there was a plugin called ***ZotFile*** while allowed me to name the stored files according to my desired collections.

11.3 Citations

There is much debate in the internet about which citation style is more appropriate for each research article. Since this is was just a research proposal, I decided to go with the default citation style of Latex.

11.4 Plagiarism detectors

I've been using plagiarism detectors even before I started academic writing. There is no proper detector that I always use. I would just Google for free plagiarism detectors and go with the top result. But since these are mostly free versions, sometimes I have to split the text into 1000 word blocks and do plagiarism checks for each block.

References

[1] Yike Liu, Tara Safavi, Abhilash Dighe, and Danai Koutra. Graph Summarization Methods and Applications: A Survey. ***ACM Computing Surveys***, 51(3):1–34, June 2018. 00055.

[2] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. ***Computer Networks and ISDN Systems***, 30(1-7):107–117, April 1998. 17889.

[3] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some Simplified NP-complete Problems. In ***Proceedings of the Sixth Annual ACM Symposium on Theory of Computing***, STOC '74, pages 47–63, New York, NY, USA, 1974. ACM. 02845 event-place: Seattle, Washington, USA.

[4] George Karypis and Vipin Kumar. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. ***SIAM Journal on Scientific Computing***, 20(1):359–392, January 1998. 05093.

[5] B. Hendrickson and R. Leland. The Chaco user's guide. Version 1.0. Technical Report SAND–93-2339, 10106339, November 1993. 00000.

[6] Mijung Kim and K. Selçuk Candan. SBV-Cut: Vertex-cut based graph partitioning using structural balance vertices. ***Data & Knowledge Engineering***, 72:285–303, February 2012. 00061.

[7] Isabelle Stanton and Gabriel Kliot. Streaming graph partitioning for large distributed graphs. In ***Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '12***, page 1222, Beijing, China, 2012. ACM Press. 00304.

[8] Hooman Peiro Sajjad, Amir H. Payberah, Fatemeh Rahimian, Vladimir Vlassov, and Seif Haridi. Boosting Vertex-Cut Partitioning for Streaming Graphs. In ***2016 IEEE International Congress on Big Data (BigData Congress)***, pages 1–8, San Francisco, CA, USA, June 2016. IEEE. 00015.

[9] Peixiang Zhao, Charu C. Aggarwal, and Min Wang. gSketch: on query estimation in graph streams. ***Proceedings of the VLDB Endowment***, 5(3):193–204, November 2011. 00052.

[10] Arijit Khan and Charu Aggarwal. Query-friendly compression of graph streams. In ***2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)***, pages 130–137, San Francisco, CA, USA, August 2016. IEEE. 00008.

[11] Nan Tang, Qing Chen, and Prasenjit Mitra. Graph Stream Summarization: From Big Bang to Big Crunch. In ***Proceedings of the 2016 International Conference on Management of Data - SIGMOD '16***, pages 1481–1496, San Francisco, California, USA, 2016. ACM Press. 00026.

[12] Milindu Sanoj Kumarage, Yasanka Horawalavithana, and D.N. Ranasinghe. An efficient query platform for streaming and dynamic natural graphs. In ***2017 IEEE International Conference on Industrial and Information Systems (ICIIS)***, pages 1–6, Peradeniya, December 2017. IEEE. 00000.

[13] Amazon. Amazon EMR Pricing. `https://aws.amazon.com/emr/pricing/`, 2019.