A Graph Mining "App-Store" for Urika-GD

Sreenivas R. Sukumar, Sangkeun Lee, Tyler Brown, Keela Ainsworth, and Seung-Hwan Lim
Computational Sciences and Engineering Division, Oak Ridge National Lab

Introduction: "Knowledge discovery" is a limitless thirst requiring the ability to integrate and link new sources, analyzing and visualizing the integrated data in creative ways. From experience with large national scale datasets and data analysis projects, we have learned that the data-infrastructure that can sustain that kind of thirst lies in between the traditional structured query language (SQL) [1] and the modern NoSQL worlds [2] – in massive heterogeneous graphs. Graphs provide a flexible way of slicing-and-dicing data in creative ways to answer questions of business value better than traditional business intelligence systems. However, data analysis on massive heterogeneous graphs has challenges scaling up to volume and heterogeneity today. Not being able to scale up for 'Big Data' is an infrastructure challenge, while not being able to analyze heterogeneous graphs is a gap that needs to be filled through design of scalable algorithms. This paper presents a solution that addresses both these challenges with the Urika-GD platform.

Graph Mining at Scale: State-of-the-art: While knowledge discovery from graph-data has been studied for a while [8], discovery from integrated information networks has seen a revitalization after the adoption of social-networks and proliferation of social-media. Recent surveys on managing and mining graph data [5] and graph algorithms [6] along with the study of laws and generating models [7] has helped design of tools such as SUBDUE [9], gSpan [10] , OddBall [11], Pegasus [12], Networkx [13], GraphLab [14,15] etc. for graph mining. However, not all of the algorithms and tools are able to scale up and handle massive datasets (in the order of terabytes). CMU's Pegasus on Hadoop [12] or Stanford's GraphLab, which can be instantiated on high performance cloud infrastructures, are restricted to mining *homogenous* graphs. On the other hand, graph-databases such as Neo4j [16], Titan [17], Trinity etc. [18] can host and retrieve massive heterogeneous graphs on commodity hardware, but do not have the data-mining functionalities of Pegasus or GraphLab.
Based on our survey of infrastructures, tools and algorithms, we identify the following scientific and technical challenges for graph-mining at scale:

(i) We do not have reliable and scalable solutions and tools for integrating, storing, retrieving and processing massive heterogeneous graph-datasets;
(ii) Properties of large real-world ad-hoc heterogeneous graphs (allowing different types of nodes and edges) are not as well understood as those of homogenous complex networks [3,4]. Existing methods assume *apriori* knowledge of a well-understood model for network formation (e.g. Barabasi-Albert [2], or Erdos-Renyi).
(iii) Analysts that work with massively parallel processing databases have difficulty with even relatively simple graph-theoretical analysis because of the extreme difficulty in writing (SQL) queries for graph patterns that are not known in advance, involve many vertices, and require approximate matching (qualities which all co-exist in graph-mining applications) [1]; and
(iv) Graph-theoretic feature identification via interactive querying algorithms on distributed storage solutions (off-the-shelf IT hardware and software) usually requires complex implementations which limit flexibility, assumes sparse relationships between entities in the graph and further, often fails to scale to the size of real-world problems.

An "App-Store" for Graphs on Urika-GD: We leverage Cray's *Urika* (Universal RDF Integration Knowledge Appliance) system [19] - a hardware platform designed specifically to provide high-speed graph-retrieval for relationship analytics. *Urika* is a massively parallel, multi-threaded, shared-memory computing device designed to store and retrieve massive graph datasets. The system can import and host massive heterogeneous graphs represented in the resource description framework (RDF) format [20] and can retrieve descriptive graph patterns specified in a SPARQL query [21]. We have leveraged this

capability to design a library of algorithms for large-scale graph analytics on *Urika*. We have implemented a toolbox of graph-theoretic algorithms on top of Cray's powerful retrieval stack that we call EAGLE – "Eagle 'Is A' Algorithmic Graph Library for Exploratory-Analysis".

EAGLE is implemented on a Python web micro-framework where the algorithms can be executed on demand as RESTful calls [22]. We have tested our library of algorithms listed in in the massively parallel multi-threaded architecture on massive heterogeneous graphs and continue to enrich the library by implementing more recent work from the literature. EAGLE is made up of the several components that we described along with their functionality below.

PLUS – Programmatic-Python Login for Urika-like SPARQL-Endpoints: This is the debug/test environment that you can build on a local machine using Apache-Jena. Once tested on Jena, it is then a matter of changing a couple of parameters to run on Urika.

FELT – Flexible Extract, Load and Transform Toolkit : Both Jena and Urika process data as RDF-triples. Unfortunately, most real world datasets live in databases or as flat files. These set of scripts loads data in common formats and when specified with a graph-model converts relational data to triples that can be analyzed using SPARQL-end points. We have a Map-Reduce Hadoop implementation for larger datasets as well.

GraphIC –Graph Interaction Console: This is the light weight graph-browser that retrieves graphs from end-points accessible through PLUS. We have tested GRAPHIC to work on display devices from IPads to ORNL EVEREST.

EAGLE – Command line: This is the set of "lego block" python scripts of popular graph-theoretic algorithms (graph summarization, centrality, Page Rank, clustering, pattern discovery etc.).

PAUSE: Predictive Analysis using SPARQL-Endpoints: A toolkit to analyze multi-structure data (numeric – data integrated with domain knowledge). PAUSE contains implementations for similarity analysis, link prediction, simultaneous feature sub-setting and feature matching.

KENODES: Knowledge Extraction using Network-Oriented Discovery Enabling System: This "app" exploits the semantic power of a machine like Urika to reason and interpret knowledge bases and is a first step in automated hypothesis generation.

We observed that the specialized hardware of Urika-GD and EAGLE middleware supporting SPARQL queries provides us with distinct advantages over conventional computer systems in performing relationship analytics. Not only are we able to compute and retrieve graphs several orders faster, we are able to compute metrics on nodes and edges that we could not before due to memory constraints of distributed graph processing. We are able to handle data sizes that are 1000 times bigger than a desktop computer for the reasonable latency that is required for interactive analytics. We prove that although Urika-GD is a machine designed for interactive graph pattern retrieval, Urika-GD good for graph mining tasks and compared to state-of-the-art large scale graph mining with Carnegie Mellon's Pegasus and Berkley's GraphX.

References

1. Celko, Joe. *SQL for Smarties: advanced SQL programming*. Vol. 2. Morgan Kaufmann, 1995.
2. Stonebraker, Michael. "SQL databases v. NoSQL databases." *Communications of the ACM* 53, no. 4 (2010): 10-11.
3. Sun, Yizhou, and Jiawei Han. "Mining Heterogeneous Information Networks: Principles and Methodologies." *Synthesis Lectures on Data Mining and Knowledge Discovery* 3.2 (2012): 1-159.
4. Albert, Réka, and Albert-László Barabási. "Statistical mechanics of complex networks." *Reviews of modern physics* 74.1 (2002): 47.
5. Aggarwal, Charu C., and Haixun Wang, eds. *Managing and mining graph data*. Vol. 40. Springer, 2010.
6. Even, Shimon. *Graph algorithms*. Cambridge University Press, 2011.
7. Chakrabarti, Deepayan, and Christos Faloutsos. "Graph mining: Laws, generators, and algorithms." *ACM Computing Surveys (CSUR)* 38, no. 1 (2006): 2.
8. Washio, Takashi, and Hiroshi Motoda. "State of the art of graph-based data mining." *ACM SIGKDD Explorations Newsletter* 5, no. 1 (2003): 59-68.
9. Holder, Lawrence B., Diane J. Cook, and Surnjani Djoko. "Substructure Discovery in the SUBDUE System." *KDD Workshop*. 1994.
10. Yan, Xifeng, and Jiawei Han. "gspan: Graph-based substructure pattern mining." *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*. IEEE, 2002.
11. Akoglu, Leman, Mary McGlohon, and Christos Faloutsos. "Oddball: Spotting anomalies in weighted graphs." In *Advances in Knowledge Discovery and Data Mining*, pp. 410-421. Springer Berlin Heidelberg, 2010.
12. Kang, U., Charalampos E. Tsourakakis, and Christos Faloutsos. "Pegasus: A peta-scale graph mining system implementation and observations." In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pp. 229-238. IEEE, 2009.
13. Hagberg, Aric, Pieter Swart, and Daniel S Chult. *Exploring network structure, dynamics, and function using NetworkX*. No. LA-UR-08-05495; LA-UR-08-5495. Los Alamos National Laboratory (LANL), 2008.
14. Low, Yucheng, et al. "Graphlab: A new framework for parallel machine learning." *arXiv preprint arXiv:1006.4990* (2010).
15. Low, Yucheng, et al. "Graphlab: A distributed framework for machine learning in the cloud." *arXiv preprint arXiv:1107.0922* (2011).
16. Partner, Jonas. *Neo4j in Action*. O'Reilly Media, 2013.
17. Titan: Distributed Graph Database on Github: http://thinkaurelius.github.io/titan/ : Accessed – Aug 18, 2013.
18. Shao, Bin, Haixun Wang, and Yatao Li. "The Trinity graph engine." *Microsoft Research* (2012).
19. Cray Yarcdata *Urika* appliance." [Online]. Available: http://www.cray.com/products/Urika.aspx
20. Powers, Shelley. *Practical RDF*. O'Reilly, 2009.
21. DuCharme, Bob. *Learning Sparql*. O'Reilly, 2011.
22. Fielding, Roy T., and Richard N. Taylor. "Principled design of the modern Web architecture." *ACM Transactions on Internet Technology (TOIT)* 2, no. 2 (2002): 115-150.