# Database Research Challenges and Opportunities of Big Graph Data

Alexandra Poulovassilis

London Knowledge Lab, Birkbeck, University of London, UK
ap@dcs.bbk.ac.uk

## 1 Overview

Large volumes of graph-structured data are becoming increasingly prevalent in areas such as

- social and professional network analysis
- recommendation services, such as product advertisement, news and media alerts, learning resource recommendation, itinerary recommendation
- scientific computing: life and health sciences, physical sciences
- crime investigation and intelligence gathering
- telecoms network management, for dependency analysis, root cause analysis, location-based service provision
- linked open data
- geospatial data
- business process management: logistics, finance chains, fraud detection, risk analysis, asset management
- organization management

Graph-structured data differs from other "big" data in its greater focus on the relationships between entities, regarding these relationships as important as the entities themselves, and allowing the possibility of modelling the attributes of relationships just as for entities, specifying constraints on relationships, and undertaking querying, analysis and reasoning over relationships.

Processing that may be undertaken with graph data includes on-line transaction processing and querying of the data on the one hand [23], and more computationally- and data-intensive off-line search, mining and analytics on the other [10].

Graph-oriented data processing algorithms may be applied to data that is stored in conventional databases, in NoSQL databases, or in specialised triple stores or graph databases. There are potentially several advantages in storing graph-structured data in specialised databases in order to undertake graph-oriented data processing: more natural support for graph data modelling [8], support for graph-oriented query languages that are better suited to formulating and optimising graph queries [25,13,2,4]; graph-specific storage and indexing for fast link and path traversal (e.g. as in Neo4J [7], DEX [18], GRAIL [26], SCARAB [14], SAINT-DB [21]), and in-database support for core graph algorithms such as subgraph matching, breadth-first/depth-first search, path finding, shortest path.

G. Gottlob et al. (Eds.): BNCOD 2013, LNCS 7968, pp. 29–32, 2013.

Beyond centralised architectures, in the area of distributed graph data processing the aim is to handle larger volumes of graph data than can be handled on a single server, with the goal of achieving horizontal scalability. Approaches include systems that provide distributed graph processing over MapReduce-based frameworks, such as Pegasus [15]; BSP (Bulk Synchronous Processing) based systems such as Pregel [17] and Giraph [3]; systems targeting distributed online processing of *ad hoc* graph queries, such as Horton [23] and Trinity.RDF [28]; and systems such as the Aurelius Graph Cluster which provide a set of technologies targeting the full range of distributed OLTP, querying and analytics [1]. In general, distributed graph processing requires the application of appropriate partitioning and replication strategies to the graph data so as to maximise the *locality* of the processing i.e. minimise the need to ship data between different network nodes.

To determine which architectures, storage and indexing schemes, computational models, algorithms, and partitioning/replication strategies are best suited to which scenarios, new benchmarks are being developed with the aim of providing comparative performance tests for graph data processing [9,20,6].

The recent Big Graph Data Panel at ISWC 2012 [5] discussed several technical challenges arising from big graph data, particularly as relating to the Semantic Web: the need for parallelisation of graph data processing algorithms when the data is too big to handle on one server; the need to understand the performance impact on graph data processing algorithms when the data does not all fit into the total main memory available and to design algorithms explicitly for these scenarios; and the need to find automated methods of handling the heterogeneity, incompleteness and inconsistency between different big graph data sets that need to be semantically integrated in order to be effectively queried or analysed.

In relation to this last point, the explicit modelling and presence in graph data of relationships between entities does provide additional means of identifying and resolving inconsistencies, through following and checking different paths between graph nodes. The explicit representation of relationships in graph data may also have implications on the required levels of consistency in certain usage scenarios, which may be more stringent than for more entity-oriented data given that the temporary presence/absence of an edge in a graph may have a large impact on query results such as reachability or shortest path. Providing the necessary levels of consistency may in turn may have performance implications on the whole workload handling.

Other challenges include: developing heuristics to drive the partitioning of large-scale dynamic graph data for efficient distributed processing, given that the classical graph partitioning problem is NP-Hard [19]; devising new semantics and algorithms for graph pattern matching over distributed graphs, given that the classical subgraph isomorphism problem is NP-complete [16]; developing query formulation and evaluation techniques to assist users querying of complex, dynamic or irregular graph data, such that users may not be aware of its full structure c.f. [11,22]; achieving scalable inferencing over large-scale graph data [24,12]; analysing uncertain graph data [29,27]; enriching of graph

data with additional inferred attributes and relationships (e.g. in a social networking setting, inferring information about peoples' interests, knowledge, skills, and social interactions); supporting users' visual exploration of large-scale graph data, and of query and analysis results; and developing algorithms for processing high-volume graph data streams.

## 2 Panel Discussion

Issues to be discussed in the panel include:

1. What are novel and emerging Use Cases that are generating large volumes of graph data?
2. How does "big" graph data differ from other graph data? Is there a spectrum of increasing volume, velocity and variety; or is there a paradigm shift?
3. Is there a fundamental separation between on-line query processing and large-scale analysis of graph data, or there is there an overlap between them? Is there a need for different architectural approaches for these two aspects of graph data processing or can "one size fit all"?
4. What graph data processing is provided more effectively by special-purpose triple stores or graph databases compared to more general-purpose databases?
5. What processing can be done well with "big" graph data now, what can be done less well, and what are the research challenges and opportunities?

## References

1. Aurelius, `http://thinkaurelius.com`
2. Cypher, `http://docs.neo4j.org/chunked/milestone/cypher-query-lang.html`
3. Giraph, `https://github.com/apache/giraph`
4. Gremlin, `https://github.com/tinkerpop/gremlin/wiki/`
5. ISWC 2012 Big Graph Data Panel (2012), `http://semanticweb.com/video-iswcs-big-graph-data-panels_b34112`
6. Linked Data Benchmark Council, `http://www.ldbc.eu/`
7. Neo4j, `http://neo4j.org`
8. Angles, R., Gutierrez, C.: Survey of graph database models. ACM Comput. Surv. 40(1) (2008)
9. Dominguez-Sal, D., Urbón-Bayes, P., Giménez-Vañó, A., Gómez-Villamor, S., Martínez-Bazán, N., Larriba-Pey, J.L.: Survey of graph database performance on the HPC scalable graph analysis benchmark. In: Shen, H.T., Pei, J., Özsu, M.T., Zou, L., Lu, J., Ling, T.-W., Yu, G., Zhuang, Y., Shao, J. (eds.) WAIM 2010. LNCS, vol. 6185, pp. 37–48. Springer, Heidelberg (2010)
10. Faloutsos, C., Kang, U.: Mining billion-scale graphs: Patterns and algorithms. In: SIGMOD 2012, pp. 585–588 (2012)
11. Fernandez, M., Suciu, D.: Optimizing regular path expressions using graph schemas. In: ICDE 1998, pp. 14–23 (1998)
12. Haffmans, W.J., Fletcher, G.H.L.: Efficient RDFS entailment in external memory. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM-WS 2011. LNCS, vol. 7046, pp. 464–473. Springer, Heidelberg (2011)

13. Harris, S., Seaborne, A.: SPARQL 1.1 Query Language. In: W3C Recommendation (March 21, 2013)
14. Jin, R., Ruan, N., Dey, S., Xu, J.Y.: Scaling reachability computation on large graphs. In: SIGMOD 2012, pp. 169–180 (2012)
15. Kung, U., Tsourakakis, C.E., Faloutsos, C.: Pegasus: A peta-scale graph mining system - implementation and observations. In: International Conference on Data Mining 2009, pp. 229–238 (2009)
16. Ma, S., Cao, Y., Fan, W., Huai, J., Wo, T.: Capturing topology in graph pattern matching. PVLDB 5(4) (2012)
17. Malewicz, G., Austern, M.H., Bik, A.J.C., Dehnert, J.C., Horn, I., Leiser, N., Czajkowski, G.: Pregel: a system for large-scale graph processing. In: SIGMOD 2010, pp. 135–146 (2010)
18. Martinez-Bazan, N., Aguila-Lorente, M.A., Muntes-Mulero, V., Dominguez-Sal, D., Gomez-Villamor, S., Larriba-Pey, J.L.: Efficient graph management based on bitmap indices. In: IDEAS 2012, pp. 110–119 (2012)
19. Mondal, J., Deshpande, A.: Managing large dynamic graphs efficiently. In: SIGMOD 2012, pp. 145–156 (2012)
20. Morsey, M., Lehmann, J., Auer, S., Ngonga Ngomo, A.-C.: DBpedia SPARQL Benchmark – Performance Assessment with Real Queries on Real Data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 454–469. Springer, Heidelberg (2011)
21. Picalausa, F., Luo, Y., Fletcher, G.H.L., Hidders, J., Vansummeren, S.: A structural approach to indexing triples. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 406–421. Springer, Heidelberg (2012)
22. Poulovassilis, A., Wood, P.T.: Combining approximation and relaxation in semantic web path queries. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 631–646. Springer, Heidelberg (2010)
23. Sarwat, M., Elnikety, S., He, Y., Kliot, G.: Horton: Online query execution engine for large distributed graphs. In: ICDE 2012, pp. 1289–1292 (2012)
24. Urbani, J., Kotoulas, S., Oren, E., van Harmelen, F.: Scalable distributed reasoning using mapReduce. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 634–649. Springer, Heidelberg (2009)
25. Wood, P.T.: Query languages for graph databases. ACM SIGMOD Record 41(1), 50–60 (2012)
26. Yildirim, H., Chaoji, V., Zaki, M.J.: GRAIL: Scalable reachability index for large graphs. PVLDB 3(1-2), 276–284 (2010)
27. Yuan, Y., Wang, G., Wang, H., Chen, L.: Efficient subgraph search over large uncertain graphs. PVLDB 4(11) (2011)
28. Zeng, K., Yang, J., Wang, H., Shao, B., Wang, Z.: A distributed graph engine for web scale RDF data. PVLDB 6(4) (2013)
29. Zou, Z., Gao, H., Li, J.: Mining frequent subgraph patterns from uncertain graph data. TKDE 22(9), 1203–1218 (2010)