Master Thesis

Using prefetching strategies to increase the performance of algorithms in graph databases

Prof. Dr. Burkhard Freitag
Chair of Information Management, University of Passau
http://www.im.uni-passau.de/

Tutor: Matthias Schmid (Matthias.Schmid@uni-passau.de)

March 21, 2019

Keywords: Database Systems, Very Large Graphs, Graph Databases, Graph Algorithms, *SQLGraph*, Prefetching

1 Introduction

Data graphs are of increasing importance in modern information systems. Therefore, an efficient method that allows to store graph data in a database is highly desirable. Unfortunately, the traditional straightforward implementation of a data graph based on edge and node tables is too inefficient. In [6] the *SQLGraph* approach to storing data graphs in a relational database has been proposed, which uses adjacency tables and can be proven to be much more efficient than the straightforward method.

In an ongoing research project at the Chair of Information Management, refined storage models and indexing schemes are investigated that are based on the *SQLGraph* proposal but can be tuned even better for some types of application. Different from other solutions, e.g. the Neo4J system [5], *SQLGraph* is based on a relational database system and therefore the graph data are not necessarily stored or available in main memory. Moreover, the graph data may be stored on a remote server. The performance of algorithms working on graph data in *SQLGraph* may therefore suffer from latency due to transfer and/or communication costs.

2 Objectives

Many graph algorithms have a rather predictable data access scheme and can therefore potentially benefit from data prefetching (see [1]). The overall objective of this thesis is to **provide prefetching to graph algorithms that work on data graphs that are stored in a relational database according to the *SQLGraph* approach**.

In particular, a collection of queries provided by the LDBC social network benchmark interactive workload [2, 3, 4, 7] have to be analyzed w.r.t. potential benefits from prefetching. Based on the results of the analysis, a prefetching-based data caching for *SQLGraph* has to be designed that can be run on client sites and can serve to increase the performance of algorithms working on graph data.

The thesis consists of a conceptual part and an implementation part. The implementation shall prototypically implement the proposed prefetching-based caching method. Based on the prefetching prototype, the essential functionality as provided by the Neo4J API for data graphs stored according to the *SQLGraph* approach is to be implemented. Using this API the analyzed queries are to be implemented. Based on the query implementation the increased performance has to be confirmed. To this end the achieved performance is to be compared to that of an implementation that does not use prefetching and to an already existing implementation that is fully realized in SQL.

3 Tasks

Among the subtasks that have to be performed are:

- Study the literature (see below) and the existing *SQLGraph* prototype;
- Analyze a selection of queries defined by the LDBC SNB interactive workload w.r.t. potential benefits from prefetching;
- Design a prefetching-based data caching for *SQLGraph*;
- Implement the proposed solution as a prototype;
- Based on the prototype implement essential functionality as provided by the Neo4J API;
- Confirm the correctness and efficiency of the proposed method using the LDBC benchmark;
- Document the proposed solution;
- Demonstrate the proposed solution.

References

[1] W. Ali, S. M. Shamsuddin, and A. S. Ismail. A survey of web caching and prefetching a survey of web caching and prefetching. *International Journal of Advances in Soft Computing and its Applications*, 3, 03 2011.

[2] P. A. Boncz. LDBC: benchmarks for graph and RDF data management. In *17th International Database Engineering & Applications Symposium, IDEAS '13, Barcelona, Spain - October 09 - 11, 2013*, pages 1–2, 2013.

[3] O. Erling, A. Averbuch, J. Larriba-Pey, H. Chafi, A. Gubichev, A. Prat-Pérez, M. Pham, and P. A. Boncz. The LDBC social network benchmark: Interactive workload. In *Proc. of the 2015 ACM SIGMOD Intl. Conf. on Management of Data, Melbourne, Victoria, Australia, 2015*, pages 619–630, 2015.

[4] A. Iosup, T. Hegeman, W. L. Ngai, S. Heldens, A. Prat-Pérez, T. Manhardt, H. Chafi, M. Capota, N. Sundaram, M. J. Anderson, I. G. Tanase, Y. Xia, L. Nai, and P. A. Boncz. LDBC graphalytics: A benchmark for large-scale graph analysis on parallel and distributed platforms. *PVLDB*, 9(13):1317–1328, 2016.

[5] I. Robinson, J. Webber, and E. Eifrem. *Graph Databases*. Shroff Publishers & Distributors Pvt Ltd, 2016.

[6] W. Sun, A. Fokoue, K. Srinivas, A. Kementsietsidis, G. Hu, and G. T. Xie. SQLGraph: an efficient relational-based property graph store. In *Proc. of the 2015 ACM SIGMOD Intl. Conf. on Management of Data, Melbourne, Victoria, Australia, 2015*, pages 1887–1901, 2015.

[7] G. Szárnyas, A. Prat-Pérez, A. Averbuch, J. Marton, M. Paradies, M. Kaufmann, O. Erling, P. A. Boncz, V. Haprian, and J. B. Antal. An early look at the LDBC social network benchmark's business intelligence workload. In *Proceedings of the 1st ACM SIGMOD Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA), Houston, TX, USA, June 10, 2018*, pages 9:1–9:11, 2018.