# Characteristics of Graph Database Benchmarks

**Vedran Juričić**
Faculty of Humanities and Social Sciences, University of Zagreb
Department of Information and Communication Sciences
Ivana Lučića 3, Zagreb, Croatia
vedran.juricic@ffzg.hr

**Matea Radošević**
Faculty of Humanities and Social Sciences, University of Zagreb
Department of Information and Communication Sciences
Ivana Lučića 3, Zagreb, Croatia
matea.radosevic@ffzg.hr

***Abstract****. Many graph database management systems have been designed over the past years because of their suitability in special usage scenarios and their performance of read and write operations. To successfully implement them into an existing system or an application, users must often compare various graph database systems and their effectiveness. Because of their specific characteristics and data model, it is challenging to create a benchmark that is adequate for graph databases in required use cases and scenarios. This paper describes the common queries and operations in graph database systems and enumerates their important features. Also, it defines the desired characteristics of graph database benchmarks and describes the most popular benchmarks for different database management systems.*

**Keywords.** Graph data, graph database, benchmarking, benchmark characteristics

## 1 Introduction

Organizing data in graphs has become a common way to help analyzing data like social network data, computer network data, travelling data etc. The amount of data that are stored in graphs is huge and algorithms used for analyzing those graphs and data are pretty complex in most cases. For example, two of the most popular social networks, Facebook and Twitter, have about 400 billion and 60 billion edges, respectively (Ching et al., 2015). Graph Database Management Systems are of great help when it comes to working with data structures which are shown in the shape of graphs or some generalization of graphs.

Graphs in general are found to be difficult to work with, especially when it comes to working with enormous number of data and data connections stored in large graphs. Both researchers and the ones who are dealing with graphs and graph databases are working on many Graph Database Management Systems whose main purpose is to make working with data stored in graphs less complex and demanding. Ching et al. (2015) made scalability improvements to use it on graphs up to one trillion edges. They also introduced processing techniques for graphs, that allowed them to make graph databases applicable in new industries, and works on more efficient graph partitioning algorithms. Hong et al. (2015) designed indexes for graph data and signatures for vertices on basis of which a new strategy that uses the specific features of weighted set family and graph topology is created. In addition to this, they introduced an efficient subgraph matching algorithm to improve query performance. As one of the properties of graph databases are frequent changes that are made, Bok et al. (2017) made up with a dynamic partitioning scheme of Resource Description Framework (RDF) graph. The scheme creates clusters and subclusters that achieve load balancing, which are based on the average frequency of queries for observed server. Also, they minimized the number of edges to reduce the costs of transmission between servers.

The main purpose of this paper is to review Graph Database Management Systems benchmarking and to characterize forms which should be thought-about when benchmarking.

In this paper, important features od graph databases are explained and features that are found in large graphs. Applications of graph data and graph databases are also described as it is important to understand where graph databases are actually used and in which domains they are especially useful. In the next chapter, popular benchmarks for relational, object oriented, Extensible Markup Language (XML) and graph databases, and RDF, are presented. Next, the elements that should be considered when designing benchmark are defined. In the end, conclusions will be made regarding review of previously made researches.

## 2 Graph databases

To understand the importance of adequate benchmark, this chapter gives a brief description of the way data is

stored in graphs and explains the most important features of graph databases.

## 2.1 Important Features of Graph Databases

In graph databases, entities are stored as nodes. There are relations between entities and they are represented with edges. Not all graph databases are the same since the complexity of data that is stored effects on the complexity of graphs. However, features that are required by most popular applications are attributes, direction, node and edge labelling, multigraphs and hypergraphs (Dominguez-Sat et al., 2011).

Attributes contain additional information linked to nodes and edges, commonly string or numerical values. When taking observing relation between nodes, it can be symmetric or asymmetric. Symmetric relation means that the edge may be passed through from any of node to opposite one and if it is not symmetric, edges have both beginning and the end, so called the head and the tail. There are also undirected graphs, where undirected edge is shown as two directed edges.

Different types of nodes and edges are also common in some applications. Node and edge labelling identifies distinct kinds of relations.

Multigraphs and hypergraphs are special kinds of graphs. In multigraphs, two nodes are possibly connected by multiple edges, and in hypergraphs, edges are replaced with hyperedges (which are composed of unreasonable number of nodes).

## 2.2 Common Features Found in Huge Graphs

Data represented in graph in real life is much different than standard graphs by definition. Mostly, it is because datasets are huge and number of edges is bigger than anyone could imagine. However, characteristics that are frequently found in large graphs are that they contain more edged than it is expected to be the maximum number of edges in a graph and there is a power law distribution which means that some nodes have more connections then there are expected to occur.

Also, in huge graphs there is a small diameter which reassures a short path to most other nodes in small world property. There is a large connected component found in great fraction of nodes and nodes are grouped in clusters with big density of edges between them (Chakrabarti & Faloutsos, 2006).

## 2.3 Applications of Graph Data and Graph Databases

With the technology growth and increase in amount of information in general, the need for storing large amount of data has occurred. Both scientists and people who work in those industries find graphs and networks suitable for organizing and representing large amount of data where nodes stand for entities and edges represent relationships between those entities.

One on the most popular datasets that is considered to best fit in graphs are organizations and social groups, and with increase of usage of social media, social networks are also one of the most popular examples of graph data.

One of the most successful tools to examine the architecture of different kinds of social groups is Social Network Analysis. Social Network Analysis is focused on determining the pattern of people's interaction. It is done by following mathematical algorithms and systematic analysis of empirical data. However, it has applications in organizational behaviour, spread of contagious diseases, social support, mental health, diffusion of information and animal social organization, counter terrorism, viral marketing etc (SNA, 2020).

With the increasing use of social networks where people can interact with each other and share information, it has become important to have possibility to track all the relationships and also interactions. Interactions can also be found in library catalogues or scientific paper repositories.

World Wide web is also one of examples that can be viewed as a directed graph with its nodes and edges (Kumar et al., 2000). This graph has more than one billion nodes and is expected that the number of nodes and edges will grow exponentially with time. It is observed for many reasons, primarily mathematical, sociological and commercial.

In addition to World Wide Web, some other web pages that are organized like huge networks or graphs should be mentioned. Some of them are Wikipedia, Amazon, Google etc. which are the best example of nodes and connections between them, that can represent various of things, maybe in most researches they try to track interest of users by giving them additional information they could be interested in, followed by links where they can find those information.

Scientists dealing with new trends in data mining also find graph data and graph database features interesting (Chakrabarti & Faloutsos, 2006). They tend to analyze social roles and interactions, and graph databases help them manage data, but also graph generators create synthetic but enough realistic graphs which follow the patterns in real-world graphs.

Large studies of proteins and genetic interactions are presented with interaction networks where nodes represent proteins. An example of database which demonstrates the complexity of graph data is Protein Data Bank (PDB) and many researches on nodes and relationships between them but also roles in those protein networks can be made based on that kind of data (PDB, 2020).

Another real-life example is traffic and demonstration of cities or places people can visit. In that cases, nodes represent places, and edges show

connections between them. The understanding of graphs could help users find the best or quickest or cheapest route to plan their journey.

# 3 Popular Benchmarks

During ages, relational databases were standard and as a result, the most of database benchmarks are meant to appraise relational database queries. Those benchmarks highlight queries with selections, joins and sorting operations. As the use of graph databases is starting to be more frequent and graph databases focus on different types of queries than relational databases, benchmarks made for rational databases are not appropriate for measuring their efficiency.

## 3.1 Relational Databases

When talking about relational database benchmark, one of the most popular benchmarks are TPC-C and TPC-H, both products of TPC (Transaction Processing Performance Council), an organisation whose main activities are creating benchmarks and creating processes for monitoring benchmarks they created (TPC, 2020). TPC-C benchmark describes the whole environment in which users perform activities. It represents only five most commonly used transactions (in wholesale supply system), but is not meant for any specific industry. Although it is mostly use to represent industries that sell or distribute a product. On the other hand, TPC-H simulates decision support system that works with large amount of complex data and it represents all kind of industries.

Benchmarking tools that are based of TPC-C simulate workload of more users with one machine and most of them are made for a single database management system. Therefore, benchmark logic should be adjusted to database management system's properties or should be even implemented in it (Lima et al., 1970).

## 3.2 Object Oriented Databases

Relational databases and graph databases have many differences, but object-oriented databases and graph databases have more in common. Data stored in object-oriented databases can be understood as a graph, but with objects and relationships between them instead of nodes and edges.

One of the most common benchmarks for object-oriented databases is OO1 (Object Operation 1). The OO1 benchmark evaluation includes measures for inserting objects (adding a set of objects and relations between them to database), looking up objects (finding set of object for specific object identifier) and traversing connections between them (performing 7-hop operation that starts at any given node). It is designed so it can be applied to object-oriented, relational, network or hierarchical database systems and even some custom application-specific database systems (Cattell & Skeen, 1992). When working with object-oriented databases, OO1 determines dataset that consists of one type of object with firmed number of edges. Graphs that are made of those kinds of objects are mostly regular.

OO7 is also commonly used benchmark for object-oriented databases. The database consists of three different types of objects that form a tree whose depth is seven and all objects have firm number of relations between them. OO7 contain traversal queries and general queries. It is more complex and thorough then OO1, and it was designed to support multiuser operations (Carey et al., 1993).

Even though object-oriented databases form graphs, those graphs are much different than typical data that is stored in graph databases. As mentioned before, graphs in graph databases are mostly irregular, as the number of nodes can vary a lot and they are also creating clusters with small diameters. When analyzing graph data, relationships are of great interest (shortest path, patterns etc.), and in object-oriented databases more attention is given to objects and their attributes.

## 3.3 XML Databases

XML databases are another type of databases which go after model that relate entities. In XML databases, data and attributes form a tree. XMark is a popular benchmark for XML databases that is designed by taking XML standards in consideration. The queries round up basics of XML processing in database management systems. Some of them are querying for NULL values, full-text search, sorted data, path expressions etc (Schmidt et al., 2002).

XML databases and graph databases have some similarities, but XML forms trees which are only one subbranch of graphs.

## 3.4 Resource Description Framework

Web researchers, developers and scientists introduces a language to represent metadata on the Web, and it is known as Resource Description Framework (RDF). From database perspective, RDF is considered to be an extension of data models used in database management systems, mostly in graph database models, as a collection of conceptual tools that describe real-world entities (in most cases those entities are metadata on the Web) (Angles & Gutierrez, 2005).

Together with RDF, knowledge community has come up with SPARQL, a query language that helps describe relationships between entities. There is a benchmark for SPARQL queries, called LUMB, which is actually a benchmark for OWL (Web Ontology Language) Knowledge Base Systems that is part of W3C's Semantic Web Technology stack and includes RDF, SPARQL, etc (Guo et al., 2005).

## 3.5 Graph databases

Since graph databases are much younger than for example relational databases, there are less benchmarks for graph database models. The most popular benchmark meant for measuring efficiency of graph data is High Performance Computing (HPC) Scalable Graph Analysis Benchmark (Bader et al., 2009).

### 3.5.1 HPC Scalable Graph Analysis Benchmark

HPC Scalable Graph Analysis Benchmark was designed by scientists and researchers in collaboration with developers from several companies. They tried to gather the most commonly used and therefor the most representative operations, like graph loadings, graph traversals and short navigations (Dominguez-Sal et al., 2010).

This benchmark is made of different operations over graph data and it simulates a power law distribution. The operations are: insertion of a graph database, restoring edges whose weight is the greatest, performing k-hops and measuring the number of edges traversed per unit of time.

### 3.5.2 LinkBench

LinkBench is a new synthetic database benchmark based on the Facebook social graph. It is based on traces from making databases storing social graph data at Facebook and reflects real-world database workloads for social applications. It is known that data gathered from social networks is one of the typical data stored in graph databases.

LinkBench operates in two phases. First, it generates and bulk-loads a synthetic social graph. Then, it benchmarks the graph that is stored with generated workload of database queries and takes statistics about how efficient it is. All parts of process have many parameters that can be changed to examine the workloads that have other characteristics (Armstrong et al., 2013).

# 4 Graph Database Benchmark Evaluation

When creating a graph database benchmark, there are few elements that should be considered. Most important is the structure of graph database and graphs itself. Then, benchmark designer should also take care of fields of business that are most likely to use graph database and which queries are expected to be used in those cases. Graph structure and most frequently used graph types are presented in chapter 2, and in this chapter the other elements are described.

## 4.1 Types of Graph Operations

There are few types of operations used in fields that are most frequently expected to be organized as graphs. Those are operations that can useful in any framework. They let us get information from graph (node, value of an attribute, neighbour nodes) or they let us create, change or delete graph or part of the graph. Operations that are mentioned need to be broaden in order to be adequate for use in examples from previous chapter. Operations can be grouped by types: traversals, connected components, communities, graph analysis, centrality measures, pattern matching and graph anonymization (Dominguez-Sal et al., 2011).

One of the typical operations are traversals. These are operations that examine the neighbourhood of a node following recursive principal until assigned depth or node is hit (Ciglan et al., 2012). There is a difference in traversals in directed and undirected graph because in directed graphs the direction is set from tail to head. One of traversals is also finding shortest path or cheapest journey. k-hops are other common traversals, and by calculating k-hops is meant to find out which nodes are k nodes away from the root node.

Connected components are subsets of nodes for which there is a connection between any of two nodes. Therefore, a single node can be part only of one connected component of the graph. In many examples, to determine connected components or bridges that are crucial for making components connects, is of great significance.

Similar to connected components, communities are set of nodes that are closer to each other than to any of the nodes that are not part of the community. It is definitely often found in social networks. Subgroup could be communities that are made of clusters where hierarchy is of great importance. Leskovec et al. that the structure of large social and information networks follows the pattern of graphs but when adding more and more information, actually nodes and edges, the community properties start to fade and graph partitioning algorithms become unavoidable.

Graph analysis gives more information about topology, complexity and objects that are found in graphs. It is done to find patterns in graphs and to get more information about role of objects, in this case nodes and edges. Part of graph analysis is measuring density, diameter, degree of transitivity etc.

Centrality measures as operation give information of how a single node is imbedded in entire network. Pattern matching is an operation that recognizes patterns and helps fine homomorphisms or isomorphisms.

Graph anonymization gives a duplex graph that has much in common with the original graph. It is used when datasets are modified to hide personal information.

## 4.2 Different Query Categories

Dominguez-Sat et al. (2010) have built few categories to group different operation used in a specific database:

transformation/analysis, cascaded access, scale, attributes and result. Transformations are operations that create new types of objects or change value of an attribute. The other queries are analysis. Queries do not change the graph but they serve as access to secondary storage that is necessary because memory is often not big enough for all the results that are generated in process of querying.

Cascaded pattern assume that a query performs neighbour operations and that the depth is at least two. On the other hand, noncascade operations do not request the neighbours of observed node. Scale gives two types of queries, global and neighbourhood ones. Attributes differentiate queries by the attribute set it accesses (edge, node, mixed or no attributes accessed).

Three types of results are found: graphs, aggregated results and set. Most common output is graph which is usually part of the original graph. Aggregates are frequently histograms, while sets are made of entities or some other sets that are not graphs.

In benchmarking, queries should serve as workload of the real surrounding and therefore should be tested on the observed application. There are different graph operations and queries, but they mostly follow the same pattern. Most operations approach edges and information that are stored in them which are actually the most important information the graph has. Graph database should have ability to save different type of results in objects that are temporary to help operations proceed to the end. Some of the result should be stored in graph in order to help analyzing graphs.

Since some operations are extremely complex, it could be impossible to calculate them, so it should be taken in consideration that sometimes the approximate values could also be useful and satisfying.

## 4.3 Experiments

Dominguez-Sat et al. (2011) also claim that experimental setup and measurement make the most important parts of benchmark. It should be well defined to grant legitimate testing of solutions in different environments. As benchmarks have become more sophisticated and databases more complex, it is not enough for benchmarks to focus on response time of queries but also pricing, answer completeness etc.

New benchmarks let the sizes of datasets to be scaled so the comparison is fair. One example of scaling is algorithm presented by Leskovec et al. (2010). It calculates one specific measure on the basis of which synthetic graph that imitates the features of desired graph is created. Also, data partition, indexing, redundancy and data reorganization should be clearly defined. As it is common when working with databases, ACID (atomicity, consistency, isolation, durability) features should be respected. Experimental process is expected to minimize side-effects of other events that could affect the experiment results. It is done in three phases: computer warm-up, query generation (to define order of queries) and observational sampling procedure (sets the number of executions for queries and how they are compiled). Phases can and should be modified depending on goals and applications.

The most common measures in graph benchmark are load time, which give the time needed for loading the dataset, size of the graph. Also, query response time, that represents the time passed until the query gives the results, and throughput, which stands for ratio of number of queries that are completed and time that was required for those queries to be completes. In the end, there are price and power consumption. Price includes the costs of hardware, software, license and other costs that occur, while power consumption evaluates demands regarding the use of energy.

There are some special metrics, for example, HPC Scalable Graph Analysis Benchmark came up with measure that gives the number of traversals per second. On the other hand, Duan et al. (2011) gave another measure and called it a macroscopic metric or coherence. They worked with RDF benchmarks and their measure combines more different primitive metrics and measures the structuredness of different datasets (Songyun et al., 2011). Measures that can be determined in benchmarking are various, but which measures are about to be selected depends on benchmark and the goals. It is to be expected that benchmarks that are done inside industry would pay more attention on price (actually, cost of queries) and throughput. On the other hand, benchmarks that are done in research purposes will more likely expect to give answers on how efficient observed operations are in different environments.

| | LDBC | HPC | McColl | XGDBench | WGB | Sotirios |
|---|---|---|---|---|---|---|
| Bulk operations | | | ID | | | ID |
| Single operations | All | | | ID | All | ID |
| Shortest path | | | x | | | x |
| BFS / DFS | | x | x | | | |
| Multiple node types | x | | | | | |
| Centrality | x | x | | x | x | x |
| K-hop | | | | | x | |
| Transaction | | | x | | | |
| Scalable | x | x | x | | x | |
| Synthesized | x | x | x | | x | x |
| Directed | x | x | | | | |
| Weighted | | x | | | | |

Table 1. Benchmarks and supported features

Table 1. shows popular benchmarks and their supported characteristics or features, with emphasis on queries and data structure (Tang, 2016). The tested benchmarks include Linked Data Benchmark Council (LDBC) benchmark (Angles et al., 2014), HPC and McColl benchmark (McColl et al., 2014).

First two rows show their support for basic bulk and single operations, i.e. insert, select, delete and update. ID in cells stands for insert and delete operations.

It can be seen that the most benchmarks do not support shortest path operations, breath-first and depth-first searches. Also, they generally do not support k-hop operations. This is certainly a great disadvantage, because those operations are common in majority of graph database usages, but users are not able to measure their performance.

Centrality, a measure of node importance in a graph, is supported by all benchmarks except XGDBench, which, on the other hand, is the only one that supports transactions.

The last four characteristics refer to the type of supported data and potentials of generators of synthetic data. As it can be seen from the table, most of the observed benchmarks support generation of synthetic data and have the possibility to scale them according to the benchmark or user needs. Directed graphs are supported in two benchmarks and weighted graphs only in one benchmark.

# 5 Conclusion

In this paper, important characteristics of graphs and graph databases are presented. Also, it is shown that there exists a need for designing more specific graph database benchmark. One of the main reasons is because graphs and graph databases are used in many different industries which could gain profit out of it and improve their work.

When designing a benchmark, it should be taken in consideration that graph characteristics are important, that it is necessary to be informed about the application of database and to determine which properties are about to be measured.

Also, different existing benchmarks are presented. In future work, it is planned to examine the differences in synthetic and real-world workloads, and to compare their advantages and disadvantages, but also to give recommendations when creating synthetic workloads. When examining graphs and their structure, it is necessary to find the quickest way to determine patterns that occur more often so that communities and subgraphs can be found and maybe some insights can be made on representative subgraphs instead of the graph in whole.

# References


Angles, R., Boncz, P., Larriba-Pey, J.L., Fundulaki, I., Neumann, T., Erling, O., Neubauer, P., Martínez-Bazan, N., Kotsev, V., Toma, I. (2014) *The Linked Data Benchmark Council: A graph and RDF industry benchmarking effort. ACM SIGMOD Record*. 43. 27-31. 10.1145/2627692.2627697.

Angles, R. & Gutierrez, C. (2005) Querying RDF Data from a Graph Database Perspective. Lecture Notes in Computer Science. 3532. 346-360. 10.1007/11431053_24.

Armstrong, T., Ponnekanti, V., Borthakur, D., Callaghan, M. (2013) LinkBench: a database benchmark based on the Facebook social graph. *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 1185-1196. 10.1145/2463676.2465296.

Bader, D., Feo, J., Gilbert, J., Kepner, J., Koetser, D., Loh, E., Madduri, K., Mann, B., Meuse, T., Robinson, E. (2009) HPC Scalable Graph Analysis Benchmark v1.0. HPC Graph Analysis.

Bok K., Kim C., Jeong J., Lim J., Yoo J. (2018) Dynamic Partitioning of Large Scale RDF Graph in Dynamic Environments. In: Lee W., Choi W., Jung S., Song M. (eds) *Proceedings of the 7th International Conference on Emerging Databases.* Lecture Notes in Electrical Engineering, vol 461. Springer, Singapore

Carey, M., DeWitt, D., Naughton, J. (1993) The 007 Benchmark. *ACM SIGMOD Record*. 22. 12-21. 10.1145/170035.170041.

Cattell, R. G. G. & Skeen, J. (1992) Object operations benchmark. *ACM Trans. Database Syst*. 17, 1 (March 1992), 1–31. DOI:https://doi.org/10.1145/128765.128766

Chakrabarti, D. & Faloutsos, C. (2006) Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv*. 38, 1 (2006), 2–es. DOI:https://doi.org/10.1145/1132952.1132954

Ching, A., Edunov, S., Kabiljo, M., Logothetis, D., Muthukrishnan, S. (2015) *One trillion edges: Graph processing at facebook-scale*.

Ciglan, M, Averbuch, A., Hluchy, L. (2012) Benchmarking Traversal Operations over Graph Databases. *IEEE 28th International Conference on Data Engineering Workshops, Arlington, VA*, pp. 186-189, doi: 10.1109/ICDEW.2012.47.

Dominguez-Sal D., Martinez-Bazan N., Muntes-Mulero V., Baleta P., Larriba-Pey J.L. (2011) A Discussion on the Design of Graph Database Benchmarks. In: Nambiar R., Poess M. (eds) *Performance Evaluation, Measurement and Characterization of Complex Systems. TPCTC 2010*. Lecture Notes in Computer Science, vol 6417. Springer, Berlin, Heidelberg

Dominguez-Sal, D., Urbón-Bayes, P., Giménez-Vañó, A., Gómez-Villamor, S., Martínez-Bazan, N., Larriba-Pey, J.L., Shen, H., Pei, J., Özsu, M., Zou, L., Lu, Jiaheng, Ling, Yu, Ge, Zhuang, Yi, Shao, Jie. (2010) Survey of Graph Database Performance on the HPC Scalable Graph Analysis Benchmark. 37-48. 10.1007/978-3-642-16720-1_4.

Duan, S., Kementsietsidis, A. Srinivas, K., Udrea, O. (2011) Apples and oranges: A comparison of RDF benchmarks and real RDF datasets. *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 145-156. 10.1145/1989323.1989340.

Guo, J., Pan Z., Heflin, J. (2005) LUMB: A benchmark for OWL knowledge base systems. *Semantic Web Journal*.

Hong, L., Zou, L., Lian, X., Yu, P.S. (2015) Subgraph Matching with Set Similarity in a Large Graph Database. *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 9, pp. 2507-2521. 10.1109/TKDE.2015.2391125.

ISNA. International Network for Social Network Analysis. Retrieved from www.insna.org/what-is-sna.

Kumar, R., Raghavan, P., Rajagopalan, S., Sivakumar, D., Tompkins, A., Upfal. E. (2000) The Web as a graph. *In Proceedings of the nineteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS '00)*. Association for Computing Machinery, New York, NY, USA, 1–10. DOI:https://doi.org/10.1145/335168.335170

Leskovec, J., Chakrabarti, D., Kleinberg, J., Faloutsos, C., Ghahramani, Z. (2008) Kronecker Graphs: An Approach to Modeling Networks. *Journal of Machine Learning Research*. 11. 10.1145/1756006.1756039.

Leskovec, J., Lang, Kevin, Dasgupta, K., Anirban, Mahoney, Michael, W. (2008) Statistical properties of community structure in large social and information networks. *Proceeding of the 17th International Conference on World Wide Web 2008*, WWW'08. 10.1145/1367497.1367591.

Lima, M., Sunyé, M., Almeida, E., Direne, A. (1970) Distributed Benchmarking of Relational Database Systems. 10.1007/978-3-642-00672-2_49.

Mccoll, R., Ediger, D., Poovey, J., Campbell, D., Bader, D. (2014) A performance evaluation of open source graph databases. *PPAA 2014 - Proceedings of the 2014 Workshop on Parallel Programming for Analytics Applications*. 14. 11-18. 10.1145/2567634.2567638.

PDB. Protain Data Bank. Retrieved from www.rcsb.org/

RDF/XML Syntax Specification (Revised). Retrieved from www.w3.org/TR/REC-rdf-syntax/.

Schmidt, A., Waas, F., Kersten, M, Carey, M.J., Manolescu, I., Busse, R. (2002) XMark: a benchmark for XML data management. *In Proceedings of the 28th international conference on Very Large Data Bases (VLDB '02)*. VLDB Endowment, 974–985.

SWAT Projects – the Lehigh University Benchmark (LUMB). Retrieved from http://swat.cse.lehigh.edu/projects/lubm/

Tang, Y. (2016) Benchmarking Graph Databases with Cyclone Benchmark. Graduate Theses and Dissertations. 15820. https://lib.dr.iastate.edu/etd/15820

TPC. Transaction Processing Performance Council. Retrieved from www.tpc.org