Beyond RDBMS

Amitabha Bhattacharyya (*Author*) Asst Professor, Computer Applications Techno India University
Salt Lake, INDIA
amitabhabhattac@gmail.com
Techno India University
Salt Lake, INDIA

***Abstract*—The advantages of RDBMS model and design methodology are being utilized by industry/institutions for any software design and implementation. The future of RDBMS certainly will be the Graph Databases with NoSql methodologies, which is emerging as beyond of relational model. In this paper we will highlight all the databases evolved after RDBMS and couldn't stay in market for so long period and survey has been made to highlight those databases after RDBMS. Relational Database Management System has certain advantages like (i) Storing in Tables, Column and Rows (ii) Data Storing in Normal Form (iii) Easy to use via SQL to retrieve information via complex join operators (iv) Maintainability via Reverse Engineering (v) Indexing and quick searches. Due to this inherent features of RDBMS and SQL, it is necessary to re-engineer and compare RDBMS where there will be no NoSQL methods or paradigm shift towards semantic databases where we can avoid complex join operations. Recently, numerous software industries and research institutions are trying their old RDBMS system to be re-engineered into some other architecture via nodes, edges and relationships where different type of information can be stored easily. So, it is a big challenge for any industry and institutions how quickly they can re-engineer their old RDBMS into Graph Databases which is also called now-a-days the future of databases. In this project, it is highlighted that the importance of the re-engineering work lies in three different direction such as (i) Comparison of RDBMS with GDBMS where facebook, twitter, Amazon, Google are adopting (ii)Survey work of Graph Databases and (iii) Graph Database Models is increasingly a topic of interest. The representation of data in the form of a graph lends itself well to structured data with a dynamic schema. This article goes over current applications and implementations of graph databases, giving an overview of the different types available and their applications and beyond RDBMS. Due to wide spread of graph algorithms and models, no standard system or query languages has been defined for graph databases. Research and industry adoption will determine the future direction of graph databases. (iv) Beyond RDBMS artifacts established by industry and academics.**

***Keywords*— (*RDBMS; GDBMS AND NoSQL; SQL,Graph*)**

I. Introduction (GDBMS)

In the past few years there has been a re-emergence of interest in storing and managing graph data. So before begin with Graph Databases, we want to show how various database techniques has evolved after RDBMS in pictorial form. In academia and research, we see many new attempts at providing a database model for large graph data, particularly social graphs and the Webgraph. While, more and more commercial applications are looking towards graph databases for their dynamic schema and ease of use in storing more complex data. This paper will go through many of the current database models giving a comparison of the different design implementations and trade off. Historically the birth of graph theory is attributed to the Swiss mathematician Leonhard Euler, who first solved the Seven Bridges of Konigsberg problem in 1736. This problem introduced the concept of representing data in the form of a graph (a set of vertices or nodes with edges joining them) and determining the traversal of the graph that results in every edge being crossed only once. Some components, such as multi-leveled equations, graphics, and tables are not prescribed, although the various table text styles are provided. Graph theory translates to today's work in computational biology and social graphs with shortest path queries, clustering, community detection, and other graph algorithms. The optimization of these queries separates graph databases from the rest. The research of graph databases was popular in the early 1990s with database models like LDM, GOOD, O2, and GraphDB. However, this interest fused off with the insurgence of XML and the Internet. Not until recently have graph databases again become a topic of interest. This re-emergence is due in part to the large amounts of graph data introduced by the Web. Just this year, the first graph conference - Graph Connect 2012 [1, 2] - was held directing only on graph databases and the adoption of such models. The recent research, following the NoSQL movement, has moved away from relational databases to ones better suited for a given application. While much this movement is focused around the horizontal scalability of data with column- store and key-value-stores, the graph data model provides a greater level of data complexity in comparison. Figure 1 shows a pictorial view of beyond RDBMS development, fig 2 and fig 1 shows the categorization of NoSQL data models comparing data complexity versus data size. Graph data models provide a higher level of data complexity in return for being able to handle less data. The remainder of this paper is structured as follows, section 2 will go over related work and past surveys of of graph database models, section 3 will cover applications and types of graph data and section 4 goes through a number of graph database models, grouped by type. Last section 5 gives some comparisons of uses of graph versus relational databases.

Figure 1.

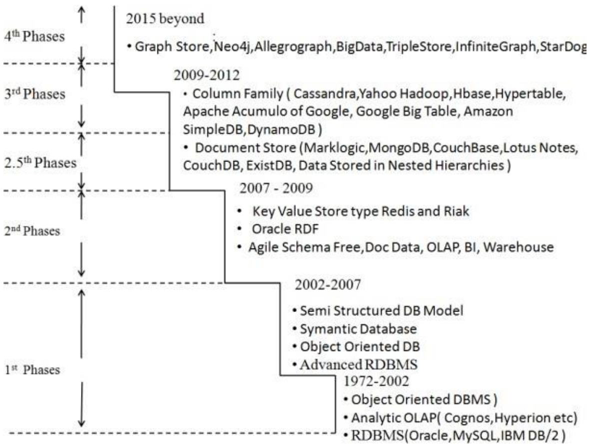


Figure 2.

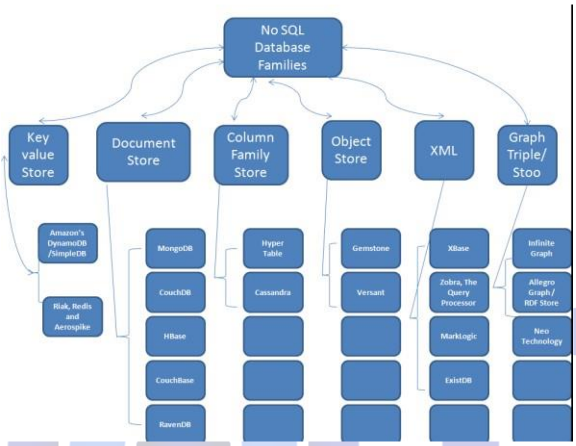


Figure 3.

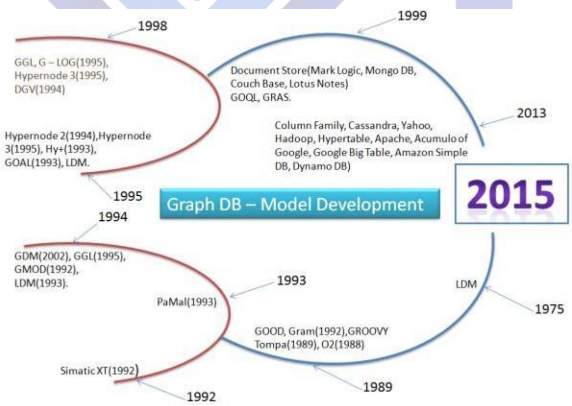


Figure 4.

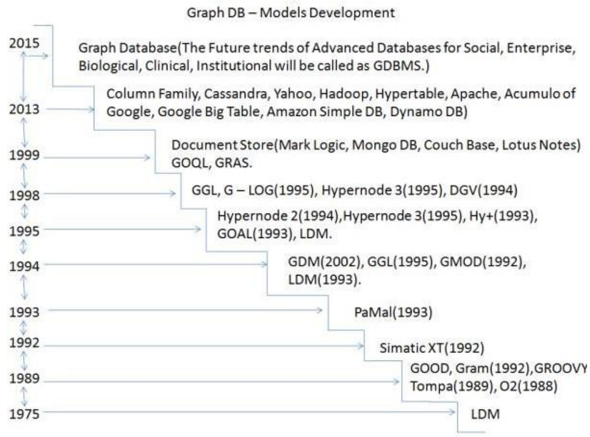
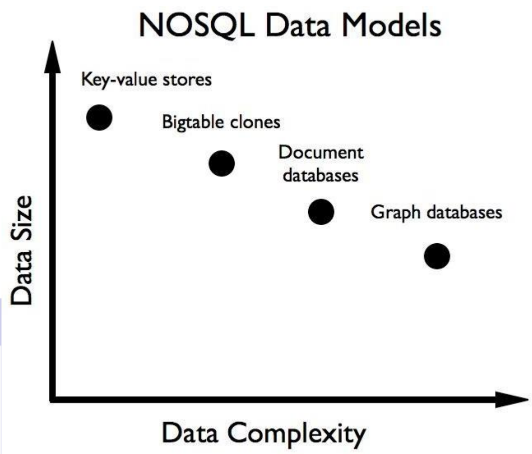


Figure 5.



II. Ease of Use

A. Related Work

Past comparisons and research of graph databases do a great job of analyzing the advantages and disadvantages. In particular Renzo Angles [#Renzo] presents a well rounded survey of graph data models and their features. The paper has gives multiple comparisons of graph data models with respect to data storing, data structure, query languages, and integrity constraints. For a survey of earlier work (pre-NOSQL) in graph databases Angles and Gutierrez [#Renzo1, #Renzo] present a survey of graph database models prior to 2002, particularly geographical, spatial and semi structured database models. It is important to notice the shift of ideas between the two papers, with respect to data schema. Older data models focused heavily on semistructured and XML data in a traditional database. Current-day data models, in contrast, focus more on providing an object-oriented or oriented, structure where the individual nodes or relations are first class priority. There is also a trend of abstraction by database models only providing API's for operation and manipulation. As many of the graph databases remain unchanged from these surveys, this paper will instead highlight more on the application of each database model and categorize them into different types.

B. Graph Applications

Some argue that most data is inherently a graph, and that all data can be stored as a graph. Using graphs to store data not only allows for a dynamic schema, but also provides representations of data not previously possible. The ability overlay different graphs (Ex. social, temporal, and spacial) on data extends the functionality of querying data. In Managing and Mining Graph Data[#springer, #Kate] we are introduced to a variety of applications for graph data, focusing on three major groups:
chemical and biological data, social networks, and the Web. This paper will focus on these three, with the addition of enterprise data via MongoDB examples and we may see the emergence of big data tool like Cytospace software for biological database, R-Programming for statistical tool for pattern searching for social data with minimum effort or any databases or Excel sheet data or from standard free datasets available, Scala, Weka tools.

C. Chemical and Biological

Chemical data is modeled as a graph by assigning atoms as nodes and bonds the edges between them. Biological data is represented the same way, only with amino acids as the nodes and links between them as the edges. This graph data is important for such operations as drug discovery and analysis. The data has many repeating node labels, so graph operations are focused at pattern recognition. Pattern recognition is done by finding frequent sub graphs of a given graph. Other operations include rank-retrieval and hopping which are used to determine chemical similarity. Modeled with a traditional database model, these operations would take a great deal longer, due to the recursive nature of traversing a graph. 'Cytospace' is the application used for this type of data via

GDBMS.

D. Social Networks

Social Networks is a very popular topic not just in society, but in graph research. Social networks, not only introduce a profound amount of data, but present large Graph data problems for the research community. These graphs, not only store nodes of people but also link nodes of multimedia, relationships, and messaging. For large social graphs we are most interested shortest path queries and clustering. These graph algorithms provide analysis of relations of two nodes and determination of communities or social networks. Currently social networking sites like Facebook do not use a graph databases. Instead they use key-value stores or column stores Cassandra (a column store similar to BigTable [#Mrigank,#Norbert, #Marek]). This is due to the sheer amount of throughput that must be handled. However, smaller scale systems or systems focused online querying like Pregel[25] provide optimized or distributed methods of analysis that support this type of data.

E. The Web

The Web, in its entirety, is essentially a graph of data and information linked together. Cudre-Mauroux[#Cudre] done the Web in terms of the Linked Data movement which supports the rapid dissemination of large-scale structured data through three principles: i) Unique Resource Identifiers (URIs) establish the creation of distinct data anywhere on the web. ii) Structured data, usually in the form of Resource Description Framework (RDF) triplets, provides a standard structure for data to be linked by. iii) Links to similar online resources connect the data to form communities or clusters of data. This massive graph of data presents applications in web search and data collection. PageRank, possibly on of the most well known secrets of Google, is a graph algorithm that analyzes Web data (pages) to determine a rank for pages by looking at how many pages are linked to it. Other important graph algorithms include web-document clustering and keyword search. Both of these algorithms aid in the searching and narrowing of data sets. Applications that deal with web data, if on a smaller scale can efficiently provide online querying of this graph-like data. For applications that focus on larger-scale graph data, online querying provides analytics and aggregates of graph data.

F. Enterprise Data

Graph databases are not limited to academia or large data graphs. Enterprise data provides perhaps the largest uptake of applications for graph database models. Modeling of data as a graph is not limited to scientific or web data; rather we can model most anything as a graph. The advantage of using graphs is the ability to represent more complex data models and support a dynamic schema. In particular, graph databases have been successful for companies that store hierarchies of product and financial and industry data[#gconnect]. The accordance of modeling data with relationships, allows for efficient restructuring as well as multiperspective querying. Graph algorithms are utilized the most, with applications such as these where data analytics are a large part of business. Data analysis or data science tools available now via R-Programming for vector add, sum, operations, pattern from excel file or database or gdbms and plot via PDF with any statistical tools and even it may fetch data from gdbms for statistical plots. Another possible application worth mentioning is the use of graph databases for bug localization [#vertexdb]. Overall there is a wide range of areas where graph data models are applicable.

G. Graph database models

There is a wide range of graph database models that have been introduced throughout the past few years. From implementations on top existing non-relational database models to graph database models build from the ground up, there is no standard graph database model on which graph algorithms are developed [#binshao, #Shefali]. Rather, each graph database is optimized for a specific set of task or queries. The problem resides in the multiple divisions of graph databases. Graph databases can focus on graph algorithms like shortest path queries and sub graph matching which require the whole graph to reside in memory and make distributed systems very difficult. On the other side of the spectrum, a graph database can focus on handling large graphs by scaling horizontally. This however makes many graph algorithms extremely inefficient or even impossible. Graphs can also focus on either online querying where low latency is required, or offline querying where larger data is handled. Graph database models are also divided by language, since no standard language has been introduced for proper graph querying. Most graphs implement their own API for operation and manipulation, in which only certain languages are provided the API in addition to the HTTP REST protocol. Some databases, more specifically known as RDF databases, support SPARQL querying which queries triple patterns against the large graph of triplets stored in the database. The following sections organize the different graph database models into categories corresponding to the data model type.

Graph Databases : The mainstream graph databases provide an object model for nodes and relationships. These graph databases focus on either RDF triplets, linked data, or relationships for storage. These databases often use direct memory links to adjacent nodes rather than requiring joins or keys lookups. AllegroGraph(2005)[#allegrograph] is a high-performance, software oriented database model that came as a precursor to the current generation of graph databases. It is implemented as an RDF databases, and serves as a reference implementation for the SPARQL query language. Implementations of geo-temporal reasoning and social network analysis extend the functionality of the database as well as a prolog extension. Allegrograph also partially enforces ACID while remaining scalable. DEX(2007)[#dex, #Norbert] is a very efficient, bitmaps-based graph database model written in C++. The focus of DEX is performance in the management of very large graphs, and even allows for the integration of various data sources. In addition to the large data capacity, DEX has a good integrity model for management of persistent and temporary graphs. Operation or core functionality like link analysis, social network analysis, pattern recognition and keyword search is done through their Java API. These core functionalities lend themselves well to applications like IMDb, on which experiments were done[#Norbert]. Neo4j(2007)[#neo4j] is a disk-based transactional graph database advertised as “The world leading graph database". It works on a network oriented model with relations as first class objects. The API is in Java, and supports Java object storage. The system is very efficient in graph traversals, however currently requires the full dataset on each node (work is being done on transparent partitioning). Neo4j also has partial ACID support and lends itself well to transactional enterprise solutions. HyperGraphDB(2010)[#hypergraphdb, #hypergraphdb1] is an open-source database focused on supporting generalized hypergraphs. Hypergraphs differ from normal graphs in their ability for edges to point to other edges. This representation is useful in the modeling of graph data for artificial intelligence, bio-informatics, and other knowledge representations. Hypergraph supports online querying with a Java API. Sones(2010)[#Sones, #Shefali] [15] is an object-oriented database written in C#. The graph database model provides its own query language based on SQL and supports a higher level of abstraction for graph queries. The model is based on weighted graphs and also has support for hypergraphs. Sones runs on a distributed file system to support scalability.

Distributed Graph Databases : Distributed Graph databases focus on distributing large graphs across a framework. Partitioning graph data is a non-trivial problem, optimal division of graphs requires finding sub graphs of a graph. For most data, the number of links or relationships is too large to efficiently compute an optimal partition; therefore most databases use random partitioning. Horton(2010)[8, 28] is a transactional graph processing framework created by Microsoft. Horton makes use of the Orleans cloud framework in order to query large distributed graphs. Instead of adopting a map/reduce architecture, Horton works with a distributed graph, passing a state machine across nodes. This allows for better ad-hoc querying in comparison to map/reduce systems. InfiniteGraph(2010)[#Infinitegraph] is a distributed-oriented system that supports large-scale graphs and efficient graph analysis. Rather than in-memory graphs, this system supports efficient traversal of graphs across distributed data stores. This works by creating a federation of compute nodes operated through their java API.

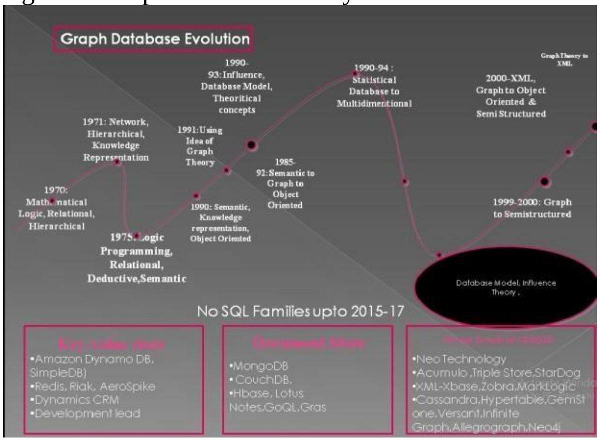
Key-Value Graph Databases : Key-value graph databases simplify the object-related model of graph databases to allow for greater horizontal scalability. These models build on, or on top of, existing key-value stores allowing for greater scalability and partitioning of graph nodes. VertexDB(2009)[#vertexdb] is a key-value disk store that makes use of TokyoCabinet. The graph database focuses on a vertex graph with added support for automatic garbage collection. CloudGraph(2010)[#cloudgraph] is an in-development, fully transactional graph database written in C#. It takes advantage of key/value pairs to store data both in-memory and on-disk. CloudGraph has also created its own graph query language (GQL). Redis Graph(2010)[#redis] is an implementation of a graph database in python using redis. Redis is a modern key-value store; the python implementation is minimalistic, creating an API in only forty lines of code. Trinity(2011)[#trinity, #bshao] is a RAM-based key value store under development by Microsoft Research. It uses message passing over a distributed system, achieving low latency queries on large distributed graphs. The benefit of in- memory key value storage can be seen with increased performance.

Document Graph Databases : Like key-value stores, document based graph databases introduce a higher level of data complexity for a given node. Orientdb(2009)[#orientdb] is a high-performance document-graph database. They make use of a novel distributed hash table algorithm in order to get greater parallelism. Another example of a document-store in graph databases is an implementation on CouchDB[#kalyani, #NRPrasanth]. This implementation makes use of the document store, in order to serve low latency queries for large graph databases. Document stores, much like key- value stores provide quick data retrieval for structured data.

SQL Graph Databases : Filament[#Filament] [#Kate] is a graph persistence library built on top of PostgreSQL. It allows SQL querying through JDBC with navigational queries for querying the graph data. G-Store(2010)[#gstore] is a prototype query language and storage manager for large graphs. It is also build on top of PostgreSQL. These implementations of graph databases are often referred to as Graph stores, for the implementation only concerns storage and retrieval of a graph data from the database, not how the data is stored.

Map/Reduce Graph Databases : To handle very large graphs, one can implement Map/Reduce functionality, in order to achieve the maximum amount of parallelism. Partitioning nodes of a graph across many machines will result in only a sizable amount of computation to be one on each machine. Pregel(2009)[#pregel] is a vertex-based infrastructure for graphs built on top of Hadoop. Hadoop, a Map/Reduce framework provides batch jobs for processing the distributed vertices with message passing. This approach only affords doing offline queries of the graph data. Phoebus(2010)[#phoebus] is another implementation of Pregel, again building on top of Hadoop in order to benefit from the Map/Reduce framework. Giraph(2011)[#giraph] also builds off of Pregel with the addition of fault tolerance. If the application coordinator has a fault, one of the available nodes will automatically become the new coordinator.

Figure 6. Graph Database Survey and Evolution



H. Comparisons

Multiple studies have been done comparing the performance of graph databases and relational ones. Graph databases like Neo4j[#neo4j] optimize for adjacency queries and graph traversal. While some operations may not be as fast as the indexing provided in a SQL database, the overall performance when doing graph-like queries will be such improved. Things to look for in graph-like queries are, lots of many to 5 many relationships, having tree like characteristics, or requiring frequent schema changes. In one comparison Neo4j and MySQL [#batra] , the authors found that graph databases did perform better than the relational model on the objective queries. However, they noted that Neo4j is not yet mature, and because there is no standard query language available it only added to this. Another paper looked at how graph databases like Neo4j performed on spacial data[#blp]. The paper found that relational databases still performed better, in all spatial queries but the ones that involved hierarchical traversal. The fact that a relational database can quickly index a coordinate location gives an advantage to relational databases. In contrast test run with a directed acyclical graph on Neo4j and MySQL[#Chad], showed a clear advantage of graph databases for structural queries. When comparisons focus on structured data with graphs that are fairly dense, relational indexing performance with joins can no longer keep up with the linked data representation in graph databases.

I. Sample survey of creating table via GraphDB

A sample survey as was done to create a database via MongoDB for developing Network database or Solar System each grid electricity generation output measurement storage databases. We choose MongoDB for special attraction how we can use NoSQL in practical term.

Figure 7. A sample Excel File

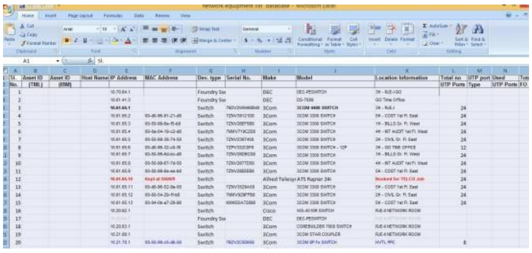


Figure 8. MongoDB Usage.

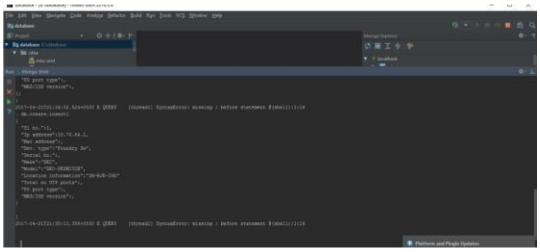
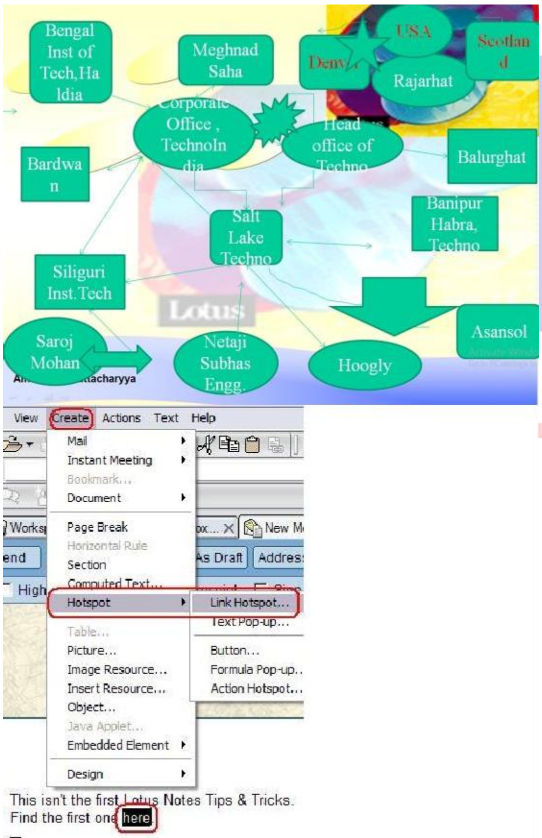


Figure 9. Integrating workflow system via document db of IBM Lotus Notes



In Excel file *.xls the information stored that we want to automate via MongoDB with pictures, connection with another distributed servers. Now to develop databases of *.xls file via MongoDB, screenshot of Excel file is being shown in this figure. MongoDB server should run in background, a special type of file to created in *.txt format to insert to MongoDB. Special File Creation for MongoDB.======{"Sl no.":1, "Ip address":10.70.64.1, "Mac address":, "Dev. type":"Foundry Sw", "Serial no.":, "Make":"DEC", "Model":"DEC-PESWITCH", "Location information":"3H-RJE-IGO" "Total no UTP ports":, "FO port type":, "MRS/IOS version":, }, { "Sl no.":2, "Ip address":10.61.41.3, "Mac address":, "Dev. type":"Foundry Sw", "Serial no.":, "Make":"DEC", "Model":"DS-7009", "Location information":"Go time office" "Total no UTP ports":, "FO port type":, "MRS/IOS version":, }, { "Sl no.":3, "Ip address":10.61.65.1, "Mac address":, "Dev. type":"Switch", "Serial no.":76DV2W9468840, "Make":"3Com", "Model":"3COM 4400 SWITCH", "Location information":"3H-RJE-I" "Total no UTP ports":24, "FO port type":, "MRS/IOS version": 3.21, }, { "Sl no.":4, "Ip address":10.61.65.2 , "Mac address":00-d0-96-91-21-d8 , "Dev. type":"Switch", "Serial no.":7ZNV39121D8 , "Make":"3Com", "Model":"3COM 3300 SWITCH", "Location information":"5H-COST 1st Fl.East" "Total no UTP ports":24, "FO port type":, "MRS/IOS version": 2.69, },... etc. For Practical for Neo4j , creating a table is much more differrent than SQL. An Example. Example shown how to code of Neo4j for database creation and queries. 1. CREATE (B001:BOOKS {title:'THE PRINCIPALS OF COMPUTER SCIENCE', PRICE:220,COPIES:20}) CREATE (A1:AUTHOR {name:'P.CHAKRABARTY'}) CREATE (A1)-[:WROTE]->(B001) Response through Bolt Response Data { "records": [], "summary": { "statement": { "text": "CREATE (B001:BOOKS {title:'THE PRINCIPALS OF COMPUTER SCIENCE', PRICE:220,COPIES:20})\nCREATE (A1:AUTHOR {name:'P.CHAKRABARTY'})\nCREATE (A1)-[:WROTE]->(B001)", "parameters": {} }, "statementType": "w", "counters": { "_stats": { "nodesCreated": 2, "nodesDeleted": 0, "relationshipsCreated": 1, "relationshipsDeleted": 0, "propertiesSet": 4, "labelsAdded": 2, "labelsRemoved": 0, "indexesAdded": 0, "indexesRemoved": 0, "constraintsAdded": 0, "constraintsRemoved": 0 } }, "updateStatistics": { "_stats": { "nodesCreated": 2, "nodesDeleted": 0, "relationshipsCreated": 1, "relationshipsDeleted": 0, "propertiesSet": 4, "labelsAdded": 2, "labelsRemoved": 0, "indexesAdded": 0, "indexesRemoved": 0, "constraintsAdded": 0, "constraintsRemoved": 0 } }, "plan": false, "profile": false, "notifications": [], "resultConsumedAfter": { "low": 1, "high": 0 }, "resultAvailableAfter": { "low": 351, "high": 0 } }, "timings": { "resultAvailableAfter": 351, "resultConsumedAfter": 1, "type": "bolt", "totalTime": 352 } }.

Practical Example of Document DB of IBM lotus Notes, Integrating all TIU Colleges Workflow, Communicating via Mailing and Approval System thereof. With Hub and Spoke server architecture we can utilise databases with any format picture, images, float, integer, char or any data without any relational concepts. We need to install Domino Server for all TIU college or any Organization and client will communicate with selected database created with document db architecture. How Moving Away from Relational Databases to Graph Databases the following pictures are self explanatory.

Figure 10. Moving away from RDBMS

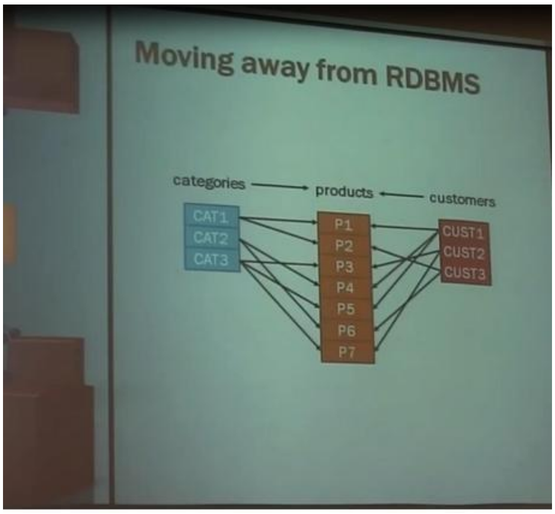
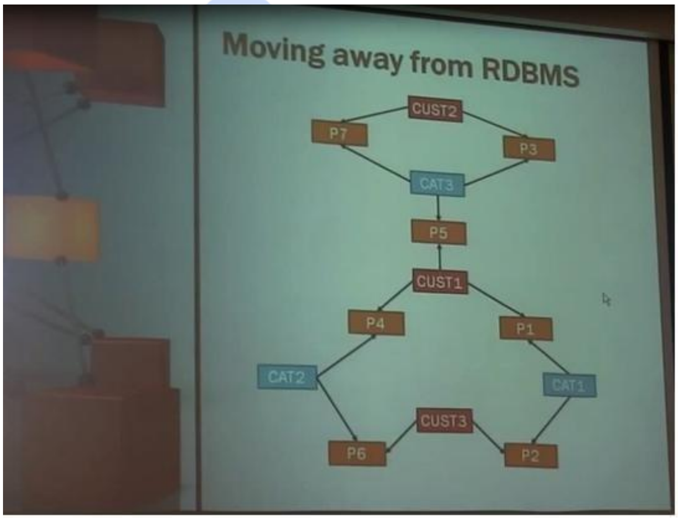


Figure 11. Moving away from Relational to GDBMS



J. Future Trends and Innovative Trends towards Beyond RDBMS

Questions regarding Beyond RDBMS relates to access storage and retrieval process via semantic methods trends. 1. Is Semantic Database is the future? 2. Is Graph Database is the "Future of Database" 3. What is the advantages of keeping GDBMS like facebook and twitter. 4. What are the operators instead of complex join operations in GDBMS, standard methodologies is yet to be explored. 5. Is the call graph databases are faster compared to Relational Database Management system? Here is the trends towards Graph Databases (figure no 2)

K. Why Paradigm shift from RDBMS to GDBMS?

Relational db-model [Codd 1970, 1983] was introduced by Codd, and highlights the concept of abstraction levels by introducing a separation between the physical and logical levels. Gradually the focus shifted to modeling data as seen by applications and users [Navathe 1992]. This was the strength of the relational model, at a time when application domains managed relatively simple data (financial, commercial and administrative applications). The relational model was a landmark development because it gave the data modeling discipline a mathematical foundation. It is based on the simple notion of relation, which together with its associated algebra and logic, made the relational model a primary model for database research. In particular, its standard query and transformation language, SQL, became a paradigmatic language for querying. The differences between graph db-models and the relational db-model are manifold. For example, the relational model is geared towards simple record- type data, where the data structure is known in advance (airline reservations, accounting, inventories, etc.). The schema is fixed, which makes it difficult to extend these databases. It is not easy to integrate different schemas, nor is it automatable. The query language cannot explore the underlying graph of relationships among the data, such as paths, neighborhoods, patterns. Semantic db-models [Peckham and Maryanski 1988] appeared as there was a need to incorporate a richer and more expressive set of semantics into the database, from a user‟s viewpoint. Database designers can represent objects and their relations in a natural and clear manner (similar to the way users view an application) by using high level abstraction concepts such as aggregation, classification, and instantiation, super-classing, attribute inheritance, and hierarchies [Navathe 1992]. In general, the extra semantics supports database design and evolution [Hull and King 1987]. A well-known example is the entity- relationship model [Chen 1976], which has become a basis for the early stages of database design. Other examples of semantic db-models are IFO [Abiteboul and Hull 1984] and SDM [Hammer and McLeod 1978]. Semantic db-models are relevant to graph db-model research because the semantic dbmodels reason about the graph-like structure generated by the relationships among the modeled entities. Object-oriented (O-O) db-models [Kim 1990] appeared in the eighties, when the database community realized that the relational model was inadequate for data intensive domains (knowledge bases, engineering applications). This research was motivated by the appearance of nonconventional database applications, involving complex data objects and complex object interactions, such as CAD/CAM software, computer graphics, and information retrieval. Sometimes it is necessary to relate the two or more table in a database. To do so we use relational database system (RDBMS). However this relational database is not suitable for web applications, computer networks, geographical structure etc., moreover in these highly connected data applications requires complex join operation which can make typical operation on this kind of data inefficient application hard to scale. To overcome this problem

Figure 12. Semantic Evolution

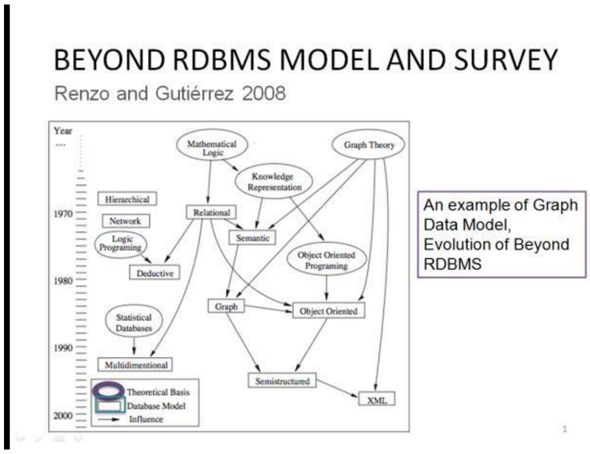
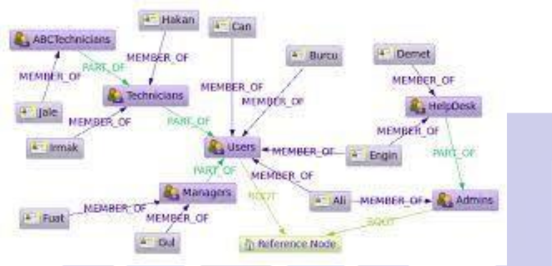
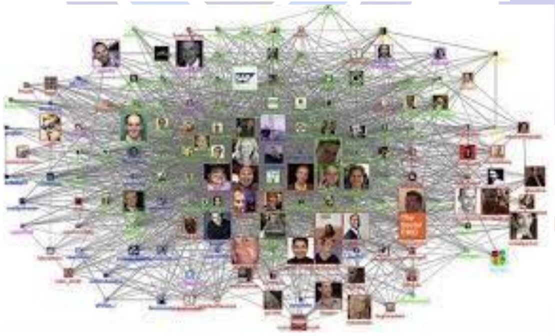
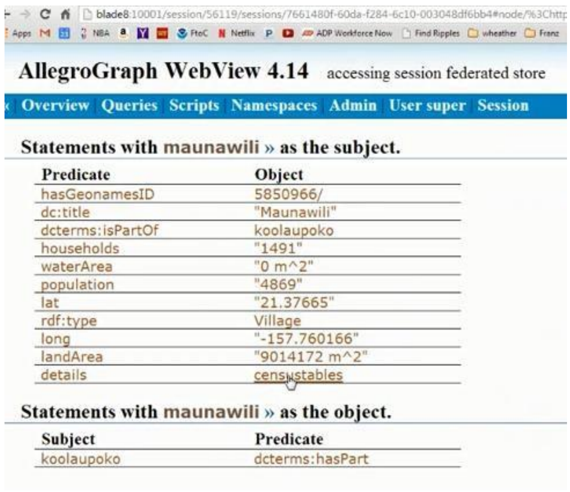
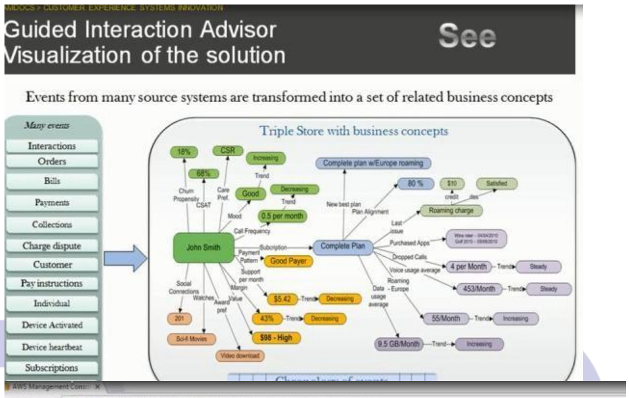


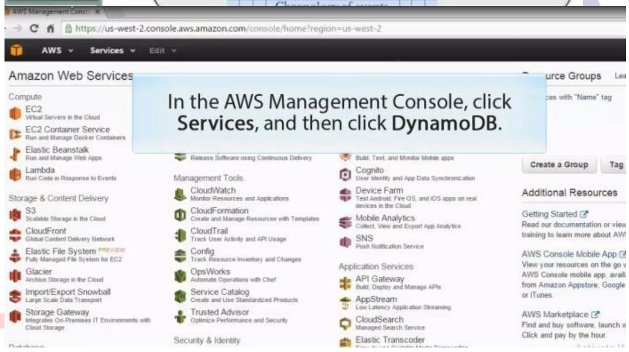
Figure 13. Facebook, twitter and many institutions trends towards GDBMS, Why? Graph Database with Nodes, Edges and relationship, inherent quality of being faster without much of thought of indexing, complex join and it"s enough which nodes connected to which nodes. Some Survey Pictures.

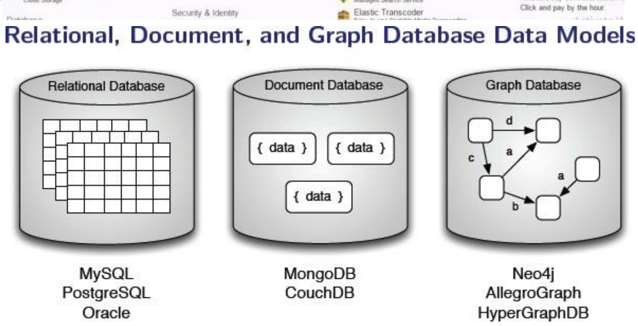


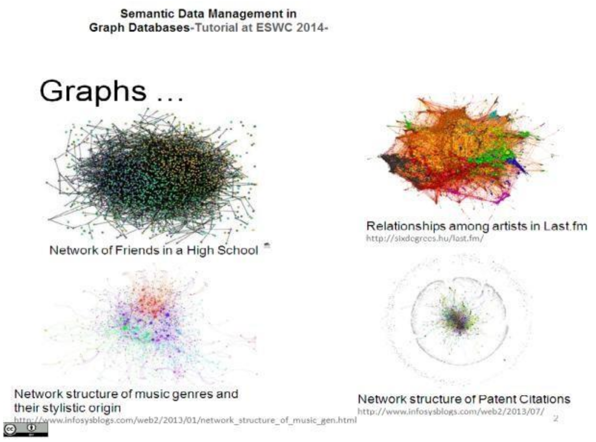


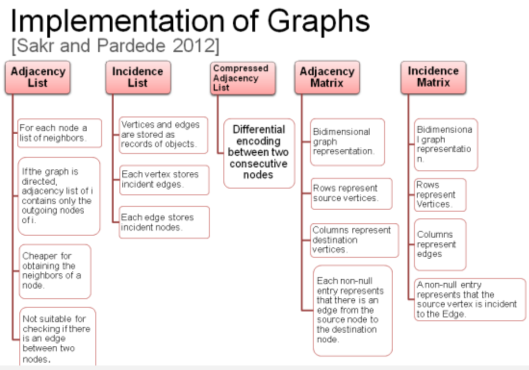


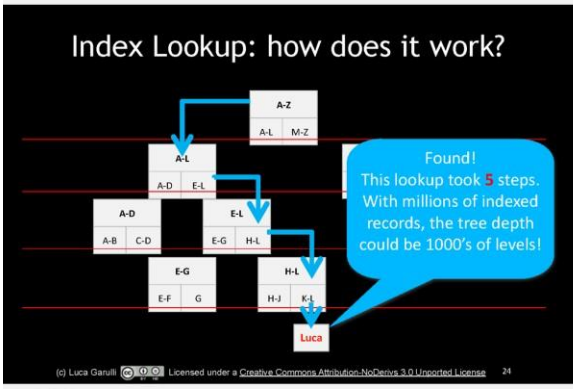












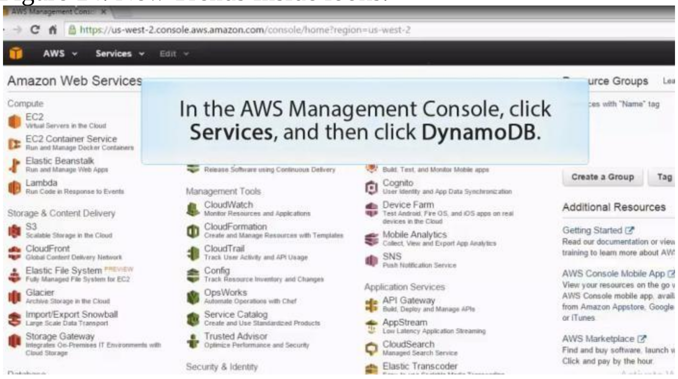
We use graph database management system (GDBMS). In GDBMS data are natively stored as graph and queries are expressed as graph traversal operation. This allows application to scale very large graph based data sets. In addition GDBMS do not rely on any schema they provide more flexible solution in scenarios where the organization of data evolves rapidly. By using graph database rather than using the relational database is more beneficial. In graph database it follows a naive approach where tuples are mapped to nodes and foreign key is mapped into edges. In this paper the network db relational database is converted into the graph database for high performance. Specifically the relational database query is converted into the graph database query. The general graph model and generic query language for graph structures.

Relational Databases: The Relational Structure Relational databases require a schema before data can be inserted. A relational database organizes data according to relations or tables, columns (attributes/properties), rows (tuples/objects).

Document Databases databases store structured documents. Usually these documents are organized according a standard (e.g. JavaScript Object Notation|JSON, XML, etc.) Document databases tend to be schema-less. That is, they do not require the database engineer to appropriately specify the structure of the data to be held in the database. MongoDB is available at http://mongodb.org and CouchDB is available at http://couchdb.org Processing JSON Documents Most document databases come with a Map/Reduce feature to allow for the parallel processing of all documents in the database. Map function: apply a function to every document in the database. Reduce function: apply a function to the grouped results of the map. M : D ! (K; V); where D is the space of documents, K is the space of keys, and V is the space of values. R : (K; V n) ! (K; V); where V n is the space of all possible combination of values. Data Management Workshop Graph databases store objects (vertices) and their relationships to one another (edges). Usually these relationships are typed/labeled and directed. Graph databases tend to be optimized for graph-based traversal algorithms. Neo4j is available at http://neo4j.org AllegroGraph is available at http://www.franz.com/agraph/allegrograph HyperGraphDB is available at http://www.kobrix.com/hgdb.jsp Graph Databases: Handling Property Graphs Gremlin is a graph- based programming language that can be used to interact with graph databases. However, graph databases also come with their own APIs. Gremlin G = (V,E) Gremlin is available at http://gremlin.tinkerpop.com.

Future work depends on categorization of data with one extra flag for big data with „weka‟ tools. It‟s the main purpose of this paper and also a glimpse of survey of gdbms. Index lookup can be grouped with B+Tree implementation with another flag attached to each node where biological, social, enterprise data, big data, corporate data, facebook/twitter data and others can be distinguished and retrieving time will be less.

Figure 14. New Trends Inside looks.



Innovative idea for classification of grouped data in a simpler way. Say 'LUCA' is being searched and found but there may be more luca in another kind of data classification which is coming in the near future and we have integrate with No-SQL, then A-Z(1 extra flag required for categorized data ranging either from 1-99 or A-Z). We need to explore more how we can put an extra flag for index lookup in Graph Database so that storage and retrieval will be easier and quick if we put the whole set of database via No-SQL paradigm. Recent trends in Graph Databases can be found in Amazon Dynamo DB look, Allegrogrpahs, MongoDB, IBM lotus Notes Db, Neo4J and books on Semantic databases.

L. Conclusions

This paper gave an overall summary of the current state of graph databases. Much of the current research in the field of applications however, in turn, the various applications found have been made with wide assortment of graph databases. In order to possibly enumerate all the different categories of graph databases, this paper depicted over many of the current graph database models being used today. Graph database models are divided by a number algorithms and paradigms which databases wish to optimize. There still does not exist a standard query language for graph databases, leading many implementations to be API only. The future of graph databases resides in the prevalence of one database over another, most likely determined by the enterprise industry and their adoption. Overall graph databases provide a much needed structure for storing data and incorporating a dynamic schema, however the research topic itself needs more structure before it can fully be adopted by industry.

Acknowledgment

I take this opportunity to express my gratitude to Prof SipraDas Bit, Prof. Durgapada Chakravarty, Prof. Prasun Ghosal, Prof. Asit Baran Bhattacharya, Dr. Anil Bikas Chaudhury and my Parents for their constant encouragement. I also thank Techno University, Salt Lake for giving us opportunity to write via IEEE, for having allowed me to pursue this paper. I also thank my well wishers and my mother for giving me good suggestions.

References

[1] Graph connect.

[2] Cloudgraph. http://www.cloudgraph.com/.

[3] R. Angles., A comparison of current graph database models., In Proceedings of the 2012 IEEE 28th International Conference on Data Engineering Workshops, ICDEW '12, pages 171(177, Washington, DC, USA, 2012.IEEE Computer Society).

[4] R. Angles, C. Gutierrez., Survey of graph database models. acm comput.surv., 40(1):1:1(1:39, february 2008)., ACM Comput. Surv., 40(1):1:1(1:39,February 2008).

[5] C. Aggarwal, H.Wang., C.C. Aggarwal and H.Wang. Managing and mining graph data, volume 40. Springer, 2010, Springer, 2010.

[6] K. Byrnep, Populating the semantic web - combining text and relational databases as rdf graphs kate byrne, Ph.D. thesis, Doctor of Philosophy Institute for Communicating and Collaborative Systems School of Informatics University of Edinburgh (2008).

[7] S. D. K. N. P. D. o. C. E. V. A. C. D. E. I. I. Mrigank Mridul,Akashdeep Khajuria, Analysis of bidgata using apache hadoop and mapreduce mrigank mridul, akashdeep khajuria, snehasish dutta, kumar n prasad.m.r dept of cse,ewit,vtu asst.prof, cse dept, ewit india indoia, International Journal of Advanced Research in Computer Science and Software Engineering 4(5), May - 2014, pp. 555-560.

[8] S. G.-V. J. N. M.-A. S.-M. Norbert Martnez-Bazan, Victor Muntes-Mulero,J.-L. Larriba-Pey., Dex: high-performance exploration on large graphs for information retrieval., in: In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, CIKM '07, pages 573-582, New York, NY, USA, 2007. ACM).,2007.

[9] L. H. Marek Ciglan, Alex Averbuch, Benchmarking traversal operations over graph databases, Tech. rep., Institute of Informatics, Slovak 24 Academy of Sciences Bratislava, Slovakia Swedish Institute of Computer Science Stockholm, https://github.com/tinkerpop/blueprints/wiki.

[10] P. Cudre-Mauroux, S. Elnikety., Graph data management systems for new application domains., in: In International Conference on Very Large Data Bases (VLDB), 2011., 2011.

[11] Vertexdb.

[12] H. W. Bin Shao, Y. Xiao., Managing and mining large graphs: systems and implementations., in: In Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, SIGMOD '12, pages 589(592,New York, NY, USA, 2012. ACM)., 2012.

[13] A. B. ShefaliPatil, Gaurav Vaswani, Graph databases- an overview, (IJC-SIT) International Journal of Computer Science and Information Technologies, Vol. 5 (1) , 2014, 657-660.

[14] Allegrograph.

[15] Dex.

[16] Neo4j.

[17] Hypergraphdb.

[18] Sones.

[19] Infinitegraph

[20] redis graph

[21] Trinity.

[22] H. W. B. Shao, Y. Li., The trinity graph engine. technical report., Tech. rep., Technical Report 161291, Microsoft Research, 2012. (2012).

[23] Orientdb.

[24] K. N. M. C. S. W. kalsatone@gmail.com, Modern graph databases models, International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 International Conference on Industrial Automation And Computing (ICIAC- 12th & 13th April 2014).

[25] I. P. D. o. C. S. S. o. E.-S. U. N.R.Prasanth, K.Arul Under graduate Student, Converting employee relational database into graph database, International Journal of Advanced Research in Computer Science &

Technology (IJARCST 2014) 514 Vol. 2, Issue 2, Ver. 3 (April - June 2014) ISSN : 2347 - 8446 (Online) ISSN : 2347 - 9817 (Print). 25

[26] Filament.

[27] G-store.

[28] A. J. B. J. C. D. I. H. N. L. Grzegorz Malewicz, Matthew H. Austern, G. Czajkowski., Pregel: a system for large-scale graph processing., in: In Proceedings of the 2010 AC SIGMOD Interna- tional Conference on Management of data, SIGMOD '10, pages 135 -146, New York, NY, USA, 2010. ACM), 2010.

[29] Phoebus.

[30] Giraph.

[31] S. Batra, C. T. C. analysis of relational, . graph databases. International Journal of Soft Computing, Comparative analysis of relational and graph databases., International Journal of Soft Computing, 2.

[32] Blp baas. nosql spatial (neo4j) versus postgis.

[33] Z. Z. X. N. Y. C. Chad Vicknair, Michael Macias, D. Wilkins., A comparison of a graph database and a relational database: a data provenance perspective., in: In Proceedings of the 48th Annual Southeast Regional Conference, ACM SE '10, pages 42:1-42:6, New York, NY, USA, 2010.ACM., 2010. 26