An overview of Graph Databases

Darshana Shimpi
Dept. of computer engineering
A.C.Patil college of engineering
Kharghar, Navi mumbai

Sangita Chaudhari
Dept. of computer engineering
A.C.Patil college of engineering
Kharghar, Navi mumbai

ABSTRACT

Graph databases are used for social networking and website link structure as graphs are used for storing connections among users. For example, social networking sites like Facebook, LinkedIn, and Twitter etc. It has more complicated networks of relationships, so to represent these databases using relational model is not efficient because of large number of table joining. Several well known graph databases are available such as Neo4j, flockdb, allegrograph, sones, trinity, hypergraphdb, infinitegraph, infogrid, orientdb, DEX. In this paper, we have presented comparison of current graph databases and it has been observed that Neo4j is the best in all current current graph databases.

General Terms

Graph databases.

Keywords

Social networking, web link structure, NoSQL.

1. INTRODUCTION

The limitations of traditional databases, in particular relational model, to cover the requirements of current application domains, has lead the development of new technologies called NoSQL (Not only SQL) databases [1]. The NoSQL taxonomy supports key-value stores, document store, BigTable, and graph databases; it can handle unstructured, unpredictable or messy data. Graph database is a NoSQL database, it uses structure as graph with nodes relationships and properties which stores and represent information. Graph databases are used in social networks, information network, technological network (e.g. internet, airline route, and telephone network), biological networks (e.g. area of genomics), and semantic web. Graph databases are a powerful tool for graph-like queries, for example computing the shortest path between two nodes in the graph. Graph database models are applied in areas where information about data interconnectivity or topology is more important as, or as important as the data itself [3]. The advantages of using graph databases are (1) natural modelling of data (2) special graph storage structure (3) efficient graph algorithms (4) schemaless (5) support for query languages and operators to query the graph structure.

Graph contains Nodes and Relationships and graph databases structure [2] is as shown below in figure 1. The simplest possible graph is a single node, a record that has named values referred to as Properties. A node could start with a single property and grow to a few million. At some point it makes sense to distribute the data into multiple nodes, organized with explicit Relationships. Relationships organize Nodes into arbitrary structures, allowing a Graph to resemble a List, a Tree, a Map, or a compound Entity – any of which can be combined into yet more complex, richly inter-connected structures.

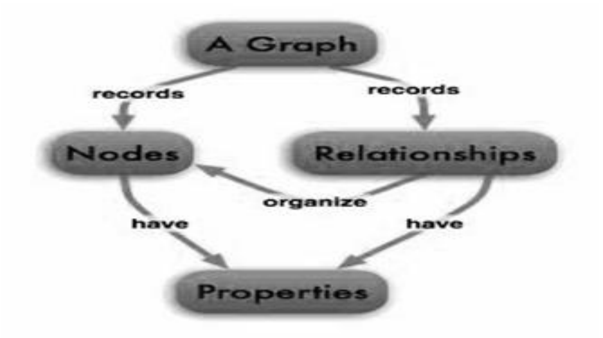


Figure 1: Graph with nodes and relationships

Generally selection of appropriate graph databases depends on the application domain, features, query languages used, data model used, advantages and disadvantages.

In this paper, we have compared current graph databases concentrating on their application, API, query facilities, reachability, usage, portability, protocol, graph type. The paper is organized as follows: In section 2, we have explained current graph databases in detail with their architecture. Section 3, presents the comparison of current graph databases. Finally, in section 4, we draw conclusions.

2. CURRENT GRAPH DATABASES

Graph databases are storage system and provide index-free adjacency. Graph databases make use of graph theory. They include nodes, edges, and properties. This model is faster for associative data sets and uses schemaless, bottom-up model for rapidly changing data.

AllegroGraph [3] is a persistent Resource Description Framework (RDF) graph database. AllegroGraph uses disk-based storage, enabling it to scale to billions of triples while maintaining superior performance. AllegroGraph supports SPARQL, RDFS++, and Prolog reasoning from Java applications. Allegrograph provides a Representational State Transfer (REST) protocol architecture Figure 2 Allegrograph is used for Geotemporal reasoning and social networking analysis.

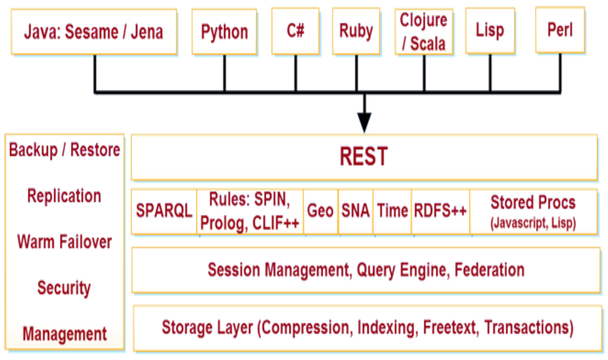

Java: Sesame / Jena
Python
C#
Ruby
Clojure / Scala
Lisp
Perl
Backup / Restore
Replication
Warm Failover
Security
Management
REST
SPARQL
Rules: SPIN, Prolog, CLIF++
Geo
SNA
Time
RDFS++
Stored Procs (Javascript, Lisp)
Session Management, Query Engine, Federation
Storage Layer (Compression, Indexing, Freetext, Transactions)

Figure 2: Architecture of Allegrograph[3]

DEX [4] is said to be a high-performance and scalable graph database and mostly used in NoSQL applications. The personal evaluation version can support up to 1 million nodes. The current version is 4.2 and it supports both Java and .NET programming. The native C++ DEX Core is the key. DEX is also portable, as only a JAR file is required to run it. Unlike Neo4J, the persisted database of DEX is a single file. As shown in figure 3 architecture of DEX, DEX Java API is easy to use, and Class Graph can provide nearly all the operations you need. DEX provides good performance in the management of very large graphs. DEX architecture has three layers as the core, the API and the application layer. The core contains three modules the GraphPool, the DbGraphmanager, the Query engine. The API provides application interface and the applications layer is used to extend core capabilities [4].

VertexDB [5] is a high performance graph database server that supports automatic garbage collection. It uses the HTTP protocol for requests and Java Script Object Notation(JSON) for its response data format and the API are inspired by the FUSE file system API plus a few extra methods for queries and queues. Its implementation is currently built on top of TokyoCabinet and libevent.Vertexdb is used for graph store.

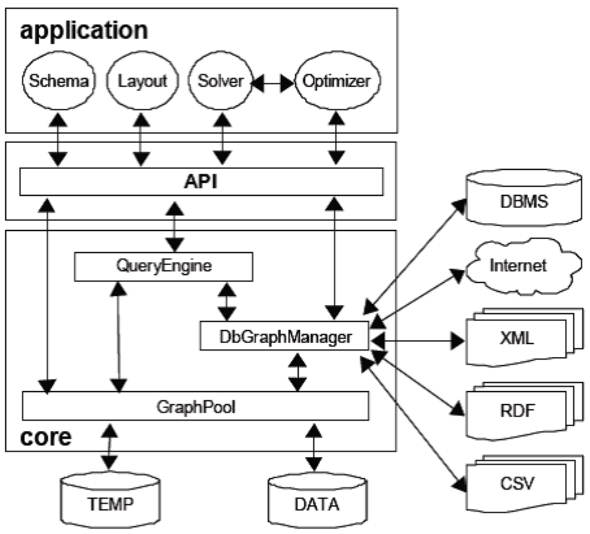


Figure 3:Architecture of DEX[4]

Infinitegraph [6] is a distributed object database with C++, java, C#, python bindings. Figure 4 shows architecture of Infinitegraph. In Infinitegraph there are four layers as user application, API, management and configuration sections and objectivity and distributed databases. Infinitegraph is used to extend business, social and government intelligence with graph analysis. Infinitegraph is built on a highly scalable, distributed database architecture where both data and processing are distributed across the network. Infinitegraph can handle large transactions efficiently. In organizations infinitegraph is used to identify and understand complex relationships between distributed data.

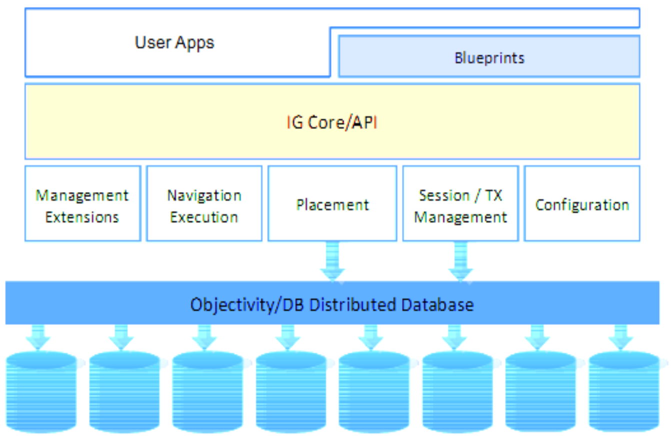
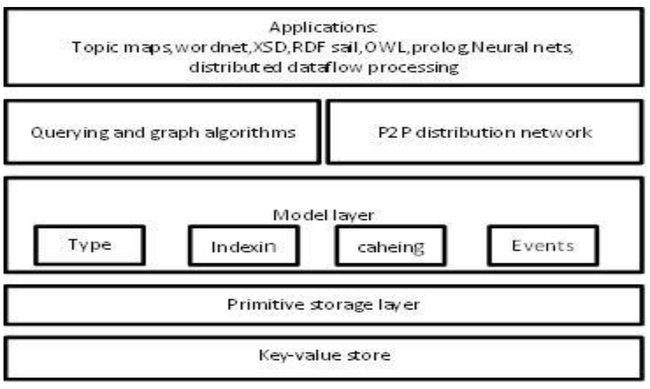


Figure 4: Architecture of Infinitegraph[6]

Hypergraphdb [7] is a general purpose, extensible, portable, distributed, embeddable, open-source data storage mechanism. While a (hyper) graph database designed mostly for knowledge representation, Artificial Intelligence (AI) and semantic web projects, it can also be used as an embedded object-oriented database for Java projects of all sizes. Figure 5 architecture of Hypergraphdb shows various layers as applications, querying, model layer, primitive storage layer and key-value store. Hypergraphdb is useful for areas like knowledge representation, artificial intelligence and bio-informatics.


Applications
Topic maps,wordnet,XSD,RDF sail,OWL,prolog,Neural nets,
distributed dataflow processing
Querying and graph algorithms
P2P distribution network
Model layer
Type
Indexin
caheing
Events
Primitive storage layer
Key-value store

.

Figure 5:Architecture of Hypergraphdb[7]

Sones [8] GraphDB has a modular structure consisting of 4 application layers as shown in figure 6. The storage engines act as the interface to different storage media. The GraphFS serializes and deserializes database objects (nodes and edges) and operates the available storage engines. The actual graph-oriented database logic as well as all functionalities specific to the database is implemented in the GraphDB. The GraphDS provides the interface for using the database. The interfaces between the application layers are generic, which makes it possible to update components separately. This graph database provides an inherent support for high-level data abstraction concepts for graphs. It defines its own query language and an underlying distributed file system.

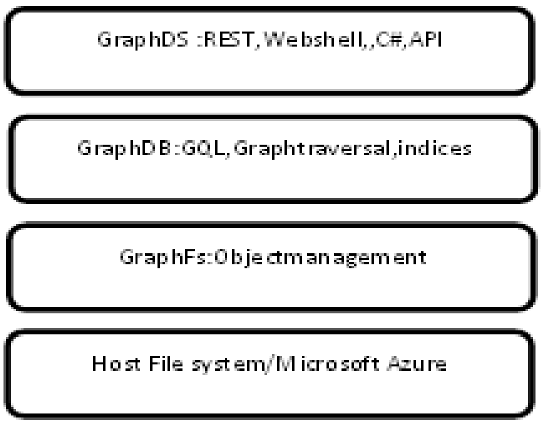

GraphDS :REST,Webshell,,C#,API
GraphDB:GQL,Graphtraversal,indices
GraphFs:Objectmanagement
Host File system/Microsoft Azure

Figure 6:Architecture of Sones[8]

Infogrid [9] is a web graph database, whose functions are oriented to web applications. InfoGrid is open source graph database developed in Java. InfoGrid Graph Database develops the graph database is a heart of InfoGrid. It can be used as a standalone graph database or in addition to the other InfoGrid projects. InfoGrid Graph Database (Grid) augments the graph database with a replication protocol, so that many distributed GraphDatabases can collaborate in managing very large graphs. InfoGrid stores provides an abstract common interface to storage technologies such as SQL databases and distributed NoSQL hash tables. InfoGrid User Interface REST-fully maps the content of a GraphDatabase to browser-accessible URLs. Viewlets allow developers to define how individual objects and sub-graphs are rendered. The project also implements a library of Viewlets, and the MeshWorld and NetMeshWorld example applications. InfoGrid Light-Weight Identity project implements user-centric identity technologies such as LID and OpenID. InfoGrid Model Library Project defines a library of reusable object models that can be used as schemas for InfoGrid applications. InfoGrid Probe Project implements the Probe Framework, which enables application developers to treat any data source on the internet as a graph of objects. InfoGrid Utilities project collects common object frameworks and utility code used throughout InfoGrid. Figure 7 shows architecture of Infogrid database.

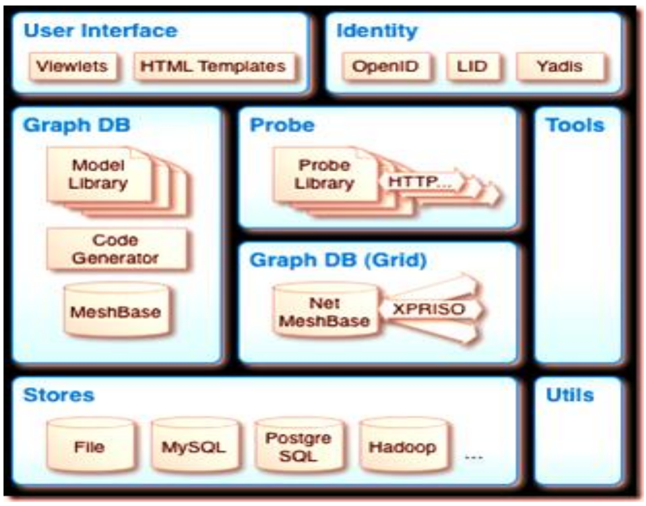


Figure 7:Architecture of InfoGrid[9]

Neo4j [10] is a high-performance, NOSQL graph database with all the features of a mature and robust database Neo4j is the most popular graph database. Neo4j is particularly developed for Java applications, but it also supports Python. Neo4j is an open source project available in a General Public License (GPL) v3 Community edition, with Advanced and Enterprise editions available under both the Advanced GPL (AGPL) v3 as well as a commercial license. The graph model in Neo4j is shown in Figure 8. It consists of (1) Property (key-value pair) can be added to both node and edge; (2) Only edges can be associated with a type, e.g., “KNOWS”; (3) Edges can be specified as directed or undirected. Neo4j uses the following index mechanism: a super referenceNode is connected to all the nodes by a special edge type “REFERENCE”. This actually allows to create multiple indexes to distinguish them by different edge types. Neo4j implements an object-oriented API, a native disk-based storage manager for graphs, and a framework for graph traversals.

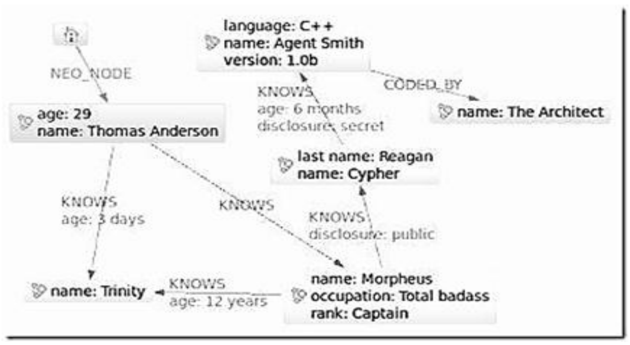


Figure 8:Graph model in Neo4j[10]

FlockDB [11] is a distributed graph database for storing adjacency lists, with goals of supporting: (1) a high rate of add/update/remove operations; (2) potentially complex set arithmetic queries; (3) paging through query result sets containing millions of entries; (4) ability to "archive" and later restore archived edges; (5) horizontal scaling including replication; (6) online data migration.

FlockDB is simpler than other graph databases such as neo4j. It scales horizontally and is designed for on-line, low-latency, high throughput environments such as web-sites. Figure 8 depicts architecture of flockdb. Twitter uses FlockDB to store social graphs (who follows whom, who blocks whom) and secondary indices. Currently the Twitter FlockDB cluster stores 13+ billion edges and sustains peak traffic of 20k writes/second and 100k reads/second.

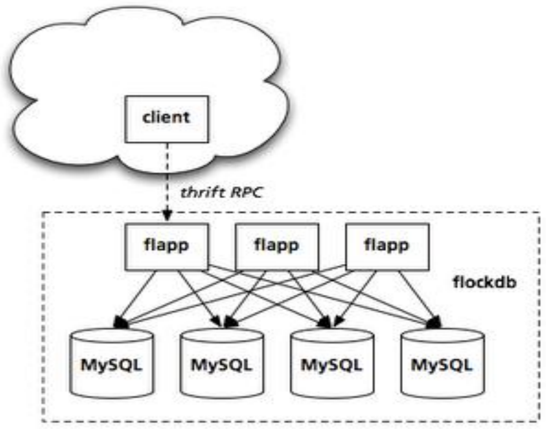


Figure 9: Structure of Flockdb[11]

Trinity [12] is a graph database and graph computation platform over distributed memory cloud. Trinity is built on top of a distributed memory storage layer called memory cloud as given in figure 10. Trinity organizes the memory of multiple machines into a globally addressable, distributed memory address space to support large graphs. Distributed key-value store layer provides key-value store interfaces to the distributed memory storage. A set of utility tools are provided by Trinity, such as fast billion node graph generator, versatile Trinity Shell and management tools. Based on these modules, graph database modules (e.g. SPARQL query module and subgraph match module) and computation platform modules are built. Trinity is used for graph store and online query processing.

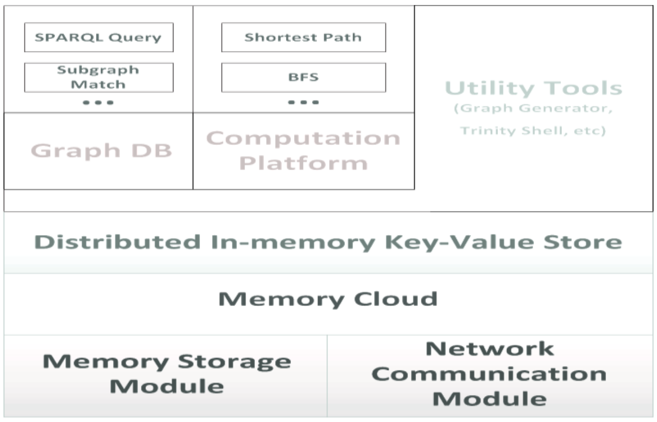


Figure 10:Architecture of Trinity[12]

Orientdb [13] graph database is an open source NoSQL database management system written in java. It is document oriented database. It supports schema-less, schema-full and schema-mixed modes. It has a strong security profiling system based on users and roles and supports SQL as a query language. OrientDB uses a new indexing algorithm called MVRB-Tree, derived from the Red-Black tree and from the B+tree; giving benefits of having both fast insertions and ultra fast lookups. Orientdb is transactional database and supports ACID transactions. It is web ready, and supports HTTP, RESTful protocol, and JSON.

G-store [14] is a prototype of a storage manager for large vertex-labelled graphs. G-Store exploits the structure of the graph to derive a data placement on disk that is optimized for access patterns found in graph queries. The placement strategy is based on a multilevel algorithm that partitions the graph into pages and arranges these pages on disk to minimize the distance on disk between adjacent vertices. G-Store has a built-in query engine that supports depth-first traversal, reachability testing, shortest path search, and shortest path tree search. The basic architecture is as shown in figure 11, there are two layers as data access layer and storage layout layer. The data access layer consists of the page layout API, the buffer manager and a library of primitive graph access patterns. The storage layout layer reads in chunks of graph data and writes them out on disk pages. G-store is used in graph storage library.

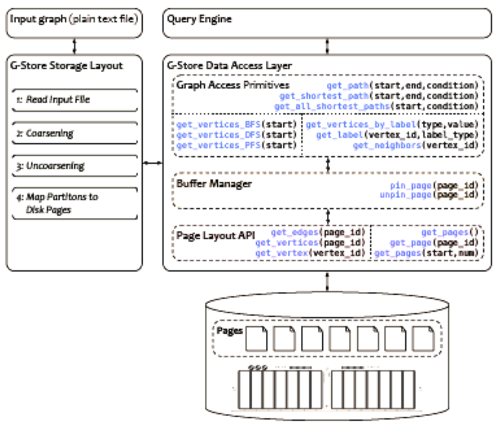


Figure 11 .G-store architecture[14]

Cloudgraph [15] is a disk- and memory-based, fully transactional .NET graph database that uses graphs and key/value pairs to store your data. The goal is to provide a fast and scalable graph database that is both easy to deploy and maintain. It traverses graph with graph query language (GQL). It also supports schemaless hypergraphs and key-value pairs. It is used for managing web data and for CLI management. Cloud graph uses API as C# and .NET.

Bigdata [16, 17] is horizontally scaled storage and computing fabric supporting optional transactions, very high concurrency, and very high IO rates. It is an open source graph database implemented in java. The Bigdata architecture provides a high performance platform for data intensive distributed computing, indexing, and a high level query on commodity clusters. It also provides SPARQL query language for fast load and query. It also supports triples. Bigdata is freely available under an open source license (GPL v2). Bigdata is used in distributed processing. Bigdata is an architecture for distributed B+ trees.

3. COMPARISON OF GRAPH DATABASE MODELS

A comparison among various graph databases is done by considering features, query languages used, API, applications, protocols used. Table I shows current graph databases (graphdb) evaluation. We consider here general features of graph databases, graph type such as simple, hypergraph or attributed, usage of the graph (for retrieval, reasoning, analysis), according to availability whether it is open source or proprietary, query languages whether SQL based or graph query language. Best graph database selection depends on the application. Allegrograph is used to store RDF triples, Neo4j and DEX graph databases are used for property graph, Hypergraphdb is used for hypergraph. Twitter uses flockdb to store social graphs, Infogrid is the web graph database, Bigdata is used for distributed processing, for storing and querying graphs G-store and vertexdb graph databases are used, Orientdb graph database is for document-oriented database, cloudgraph and trinity are as prototypes of graph databases.

Allegrograph and Infinitegraph are close source and written in java also query languages are different but they does not support portability. DEX is the high performance graph database with SQL based query language. Hypergraph, Neo4j, vertexdb, sones, orientdb, infogrid are open source graph databases but only Neo4j is having its own query language Cypher and also it supports O-O API. G-store, trinity, cloudgraph, Bigdata are close source graph databases. Trinity and Bigdata uses SPARQL query language. Among these graph databases neo4j and trinity are most widely used databases for social networking sites. Neo4j supports billions of nodes, ACID compliant transactions and multi-version consistency [18].

Most NoSQL databases horizontally scale across multiple servers by partitioning. FlockDB overcomes the difficultly of horizontally scaling the graph by limiting the complexity of graph traversal. In particular, FlockDB does not allow multi-hop graph walks, so it can't do a full explosion of parts. However, it is very fast and scalable if you only want to access first level relationships. Twitter stores many graphs of relationships between people: who you're following, who's following you, who you receive phone notifications from, and so on. Some of the features of these graphs have been challenging to store in scalable ways as we've grown. For example, instead of requiring each friendship to be requested and confirmed, you can build one-way relationships by just following other people [19]. To deliver a tweet, we need to be able to look up someone's followers and page through them rapidly. The biggest difference between FlockDB and other graph databases like Neo4j and OrientDB are graph traversal. Twitter's model has no need for traversing the social graph. Instead, Twitter is only concerned about the direct edges (relationships) on a given node (account). For example, Twitter doesn't want to know who follows a person you follow. Instead, it is only interested in the people you follow. By trimming off graph traversal functions, FlockDB is able to allocate resources elsewhere. So without traversal it is a persisted graph and not a graph database but still it is used by the biggest social networking site like twitter.

4. CONCLUSIONS

In this paper we compared current graph databases according to data modeling features. In most of the graph databases there is different data structures, query facilities in the form of APIs, and few query languages. There is a need for developing some aspects of current graph database models, in terms of standard graph database languages (for defining and querying data). An evaluation of the current graph databases shows that Neo4j is the best and most widely used graph database.

Table I:Current graphdb evaluation

Graph DB	**Protocol**	**API**	**Availability model**	**Query language**	**Reachability**	**Use**	**Graph type**	**Port ability**	**Application**
Allegrograph	REST	Java	Close source	SPARQL	Fixed length	Retrieval, reasoning, Analysis	Simple	No	Geotemporal resoning, social network analysis
DEX	-	C++, Java	Close source	SQL based	Fixed length, regular simple, shortest path	Retrieval, analysis	Attributed	Yes	Management of very large graphs
Infinitegraph	REST	Java	Close source	Gremlin blueprint support	Fixed length, regular simple, shortest path	Retrieval	Attributed	No	Business, social and government intelligence with graph analysis
Hypergraph	REST / JSON	Java	Open source	SQL style	-	Retrieval	Hyper Graphs	Yes	Knowledge representation, artificial intelligence, bio-informatics
Neo4j	REST / JSON	Java, jpython jruby	Open source	Cypher	Fixed length, regular simple, shortest path	Retrieval	Attributed	Yes	For O-O API disk based storage manager for graphs
Sones	REST / JSON	C#	Close source	SQL based graphQL	-	Retrieval, Analysis	Hyper graphs, attributed	Yes	Handling semistructured data
VertexDB	HTTP / JSON	C, C++	Close source	SQL through JDB	Fixed length, regular simple path	Retrieval	Simple	Yes	Graph store
G-store	REST	c/c++	Close source	SQL based	Fixed length, regular simple, shortest path	Retrieval	Simple	Yes	Graph storage library
OrientDB	REST / JSON	Java	Open source	SQL	Fixed length, regular simple, shortest path	Retrieval	Simple, attributed	Yes	For document oriented databases
Infogrid	REST / JSON	Java	Open source	Web user interface with html	-	Retrieval, Analysis	Simple	yes	For web applications
Cloudgraph	-	C#	Close source	GQL	-	Retrieval	Simple	Yes	Web based , CLI management, webling fast access to highly interconnected dataset
Trinity	C#language binding	C#	Close source	SPARQL	Fixed length, shortest path	Retrieval	Attributed, hypergraph	yes	Graph store, online query processing
Bigdata	-	Java	Close source	SPARQL	Fixed length	Retrieval	Simple	Yes	Distributed processing

5. REFERENCES

[1] “NoSQL Databases,”http://nosql-database.org/

[2] http://neo4j.org/learn/

[3] “Allegrograph,”http://www.franz.com/agraph/allegrograph.

[4] N. Mart´ınez-Bazan, V. Munt´es-Mulero, S. G´omez-Villamor, J. Nin, M.-A. S´anchez-Mart´ınez, and J.-L. Larriba-Pey, “DEX: High-Performance Exploration on Large Graphs for Information Retrieval,” in Proceedingsof the 16th Conference on Information and Knowledge Management (CIKM). ACM, 2007, pp. 573–582..

[5] ”vertexdb,”http://www.dekorte.com/projects/opensource/vertexdb/

[6] “Infinitegraph”,http://www.infinitegraph.com/.

[7] B. Iordanov, “Hypergraphdb: a generalized graph database,” in Proceedings of the 2010 international conference on Web-age information management (WAIM). Springer-Verlag, 2010, pp. 25–36.

[8] “Sones graph db,” http://www.sones.com/

[9] “Infogrid,” http://www.infogrid.com/

[10] “Neo4j,”http://www.neo4j.org/

[11] http://engineering.twitter.com/2010/05/introducing-flockdb.html

[12] “Trinity”,http://research.microsoft.com/en-us/projects/trinity/

[13] “Orientdb,”http://www.orienttechnologies.com/

[14] “G-store,”http”//g-store.sourceforge.net/

[15] “Cloudgraph,”http://www.cloudgraph.com/

[16] http://www.systap.com/bigdata.htm

[17] http://www.bigdata.com/whitepapers/bigdata_whitepaper_10-13-2009_public.pdf

[18] http://www.dbta.com/Articles/Columns/Notes-on-NoSQL/Graph-Databases-and-the-Value-They-Provide-74544.aspx

[19] http://engineering.twitter.com/2010/05/introducing-flockdb.html