TigerGraph

WHITE PAPER

TigerGraph and the Rise and the Future of Graph Database

WHITE PAPER

The Rise and the Future of Graph Database

A primer on the evolution of databases – from RDBMS to NoSQL and Graph

Databases play critical roles in our economy and daily life, helping us keep track of basic transactions, manage large lists such as customer contacts and inventory, and generate a range of management reports. They are also the backbone of interactive websites ranging from simple eCommerce sites to social media and search giants such as Facebook and Google.

Over the past 40 years, we have witnessed an evolution in databases driven by user need, developments in underlying hardware and software, and innovations in database concepts and functionality, culminating, as we shall see, in TigerGraph.

For much of past four decades, relational databases such as Oracle, IBM DB2, Microsoft SQL Server, and Teradata have dominated the database market due to the ease of use of SQL, well understood tabular data structures, and a well-developed ecosystem, including tools such as Tableau and Microsoft Power BI.

In the past 15 years, with more data generated than ever before, traditional relational database management systems (RDBMSs) have shown limitations in their ability to scale up to handle very large datasets, maintain performance in the face of increasing complexity and cope with organisations' need for more flexible data schemas.

Thus, NoSQL databases such as key-value/column databases (such as Redis and Cassandra) and document databases (such as MongoDB) have grown in popularity due to their ability to scale out to multiple machines and handle large datasets with diverse schema more easily.

However, even these solutions have their drawbacks as the scalability in NoSQL databases is at the cost of losing their expressive powers. Query languages and APIs supported by these NoSQL databases are not as powerful as SQL supported by traditional RDBMS vendors. NoSQL database developers often have to code in Java or Python to do things easily expressed in a high-level language such as SQL.

The answer to these problems lies in graph. The graph database is not a new concept – it has been around since the early days of modern computing – but it is only in recent years that it has reached a level of maturity where it can challenge the established order.

At its root, the graph model is more powerful than the relational model. Every table schema you design in an RDBMS has a simple equivalent in a graph database, but the reverse is not necessarily true. Every SQL query you write can be expressed in a graph query language (such as TigerGraph's GSQL), often in a more concise manner. And many high-value business problems can be easily expressed and efficiently solved in graph query languages but cannot be expressed or efficiently solved in an RDBMS, such as customer clustering, page ranking, fraud group detection, supply chain optimization, and customer journey.

The advantage for the graph database over RDBMS is due to the underlying data structure. RDBMSs store and query data as tables (comprised of columns and rows) which are physically separated on disk or in memory. In contrast, graph databases store and query data as entities and relationships (vertices and edges). The difference in structure – which gives the graph database its performance advantage – is that the edges effectively pre-join related entities so that queries can be faster at run time.

The advantage of these pre-joined links cannot be overstated. To analyze data in an RDBMS, you first have to build a temporary table of the vertices and edges you want to examine before you can even start the analysis. In a graph database, these relationships are explicit so you can skip step one and go straight to the analysis.

Tabular data structures are a two-dimensional way of representing and querying data, while a graph database offers unlimited ways to represent data in multiple dimensions. For example, geospatial data, organizational hierarchy, temporal or time-series data, and social relationships can all be naturally modeled in a graph. In fact, our brains think about the real world and make decisions more like a graph than groups of tables.

Why graph and why TigerGraph

Although implementations of graph have been around for many years, the arrival of TigerGraph in 2017 significantly changed the game.

Building a distributed commercial-grade graph database has great challenges. Distributed systems are hard to design and implement. Graph databases inherently require more random access to the data than RDBMS or NoSQL databases to 'connect the dots' and analyze relationships.

The legacy graph databases such as Neo4j were created more than 15 years ago. They suffer from technical limitations as they were designed for a single-machine architecture and cannot support high-performance computing and big datasets.

Thus historically, graph databases have been limited to small projects which worked for organizations until the advent of the big data movement started by Hadoop and other distributed NoSQL systems. TigerGraph built on the modern big data movement and had a breakthrough in the development of the first Native and Parallel graph database. Today some of the largest organizations worldwide – including seven of the top ten financial institutions, UnitedHealth Group, Jaguar Land Rover, Xandr, a division of AT&T, and others – use TigerGraph every day to solve complex data problems.

"Native" means the graph DB was built from the ground up, with compression and optimization (in storage, memory usage, and communication) purpose-built for graph processing.

"Parallel" means the graph DB is not only distributed across machines but also can take advantage of multi-cores on each machine. In TigerGraph's Native and Parallel architecture, you can imagine each vertex and edge as a parallel data storage and computation unit to achieve high-performance computing.

The impact of having developed the first Native and Parallel graph database reverberates throughout the graph space as together these two innovations represent a paradigm shift which delivers a level of maturity previously absent from the market.

Frequently Asked Questions

Primary use cases for Graph

Graph database is a horizontal technology and is used across a wide range of industries and use cases requiring advanced analytics and machine learning, including fraud detection, anti-money laundering (AML), entity resolution, customer 360, recommendations, knowledge graph, cybersecurity, supply chain, IoT, blockchain and network analysis.

Who are TigerGraph customers?

We're fortunate to work with some of the largest and most visionary customers across industries, including top Fortune 500 companies and innovative startups, and mid-size companies. TigerGraph is used by seven of the ten largest financial institutions globally and the largest power grid company, as well as a range of manufacturing, telecom, media, entertainment, retail and eCommerce companies. TigerGraph also powers the largest graph in the healthcare sector and is used in pharmaceutical companies for both clinical research and commercial operations.

What are TigerGraph's current product offerings?

TigerGraph has offered a comprehensive product portfolio since 2017, including the TigerGraph DBMS as an on-premises solution and in all three leading clouds – AWS, Azure, and GCP.

TigerGraph Version 3.1 is a native graph store supporting the property graph model and GSQL for queries. TigerGraph has developed open-source toolkits for developers, including the algorithm library, a range of connectors, and TigerGraph Cloud Starter Kits with pre-built graph schema and queries.

The on-premises enterprise edition is free up to 50 GB of raw data, and there is a free tier available on TigerGraph Cloud for both development and production.

TigerGraph Cloud is available as a SaaS offering on AWS, Azure, and GCP marketplaces, allowing customers to purchase and use TigerGraph Cloud with their committed budgets from these cloud providers.

TigerGraph features no-code migration from relational to graph and a visual query builder for designing complex analytics patterns with no coding required. It supports Python for connection to machine learning frameworks via its pyTigerGraph API and in-database machine learning via GSQL. TigerGraph's data storage architecture is suited for multi-hop (multi-level) queries utilizing a highly distributed and parallel computational implementation.

How big is the Graph market?

The graph market is growing fast, and TigerGraph is a big part of it.

Because of the superior performance and expressive power over RDBMS, we believe the addressable market for Graph is over $84B as the opportunity draws from these categories:

1. **Graph database market**
   We believe Graph DB alone can be bigger than Document DB, which includes solutions like MongoDB, because Graph solves more business problems than DocumentDB and will power more applications solving higher-value business problems.

2. **Machine Learning and AI powered by Graph**
   ML and AI are fundamentally transforming many industries. Graph complements ML/AI in many ways. Marquee customers such as Bank of America, JPMorgan Chase, and Intuit Corporation use TigerGraph to augment their machine learning capabilities and gain huge ROI by saving millions in fraud losses. We're building more in-database machine learning and AI and integrating with popular ML/AI frameworks and APIs from big cloud vendors. We believe this market alone can be even larger than the distributed graph database market itself.

3. **Graph BI market**
   Business Intelligence (BI) applications enable non-technical users to organize, analyze, and visualize data relevant to their business concerns. BI for RDBMS is a big market since it democratizes data access and reports to more users with less tech background. From the beginning, TigerGraph has offered graph visualization in its GraphStudio visual SDK. In 2020, TigerGraph released Visual Query Builder and a No-code RDBMS migration tool kit. These features were just our first steps in Graph BI.

We believe this market can be larger than the distributed graph database market as well because TigerGraph is the only distributed graph database that can power real-time graph analytics suitable for interactive and visual data analytics for end users. This capability gives us superior advantages in leading innovative products and services in the graph BI and visualization space.

4. **Vertical business applications and data/API services powered by Graph**
   We already have over 20 starter kits with graph schema, queries and sample data for businesses to quickly deploy which are application specific to their industries such as supply chain and customer-identity management. However, they are far from turnkey solutions. As we have seen more customer use cases in fraud investigation and prevention, customer identity graph, customer 360/journey, and supply chain optimization, we have begun working with partners to build applications which are nearly turnkey solutions and enable businesses to realise ROI on new projects quickly. These applications are far easier to build on Graph since data is connected and GQL is flexible and powerful. We believe this market can be larger than the distributed graph database market.

5. **GQL replacing SQL as foundation for building new applications**
   TigerGraph is on the ISO GQL committee to help to bring a single unified, powerful, and easy-to-use query language for Property Graphs to the world. We believe GQL will be the new Python for data scientists to do ML/AI and build analytics applications. Just like Java replacing Cobol as a more popular tool to build applications, we believe that in the long term GQL and Graph DB will replace SQL and RDBMS as the first choice for building new business applications and advanced analytics.

This market is massive and can be larger than all the above categories combined. Just look at how big the RDBMS vendors are (like Oracle, Microsoft, and Snowflake). We have already started working in this direction. Dr. Alin Deutch and his team's initial work has been published at SIGMOD 2021 conference (one of the most prestigious database conferences). TigerGraph already helped customers to use TigerGraph to do things previously impossible or too slow with products like Teradata or Snowflake. With GQL's progress, we believe TigerGraph will be used in even more advanced analytics across industries for use cases and workloads that are currently too slow or too expensive with RDBMS.

The above categories also indicate where we are heading with our product roadmap and company investment now and in the next few years. We're already the leading and only scalable enterprise graph database, and have a huge head start on Graph BI with our unique Visual Query Builder. Now the company is investing heavily in ML/AI powered by Graph and UI/Applications powered by Graph, in addition to continued investment and innovation in our Graph platform, especially in the Cloud.

How is TigerGraph going to grow to deliver the promises of Graph and what are our short-term goals?

Our goal is to be a $100M ARR (annual recurring revenue) company in less than three years and be IPO-ready. We started financial auditing with a Big Four accountancy firm, and we have started the company-wide governance process needed for IPO.

Summary

Graph was a paradigm shift in database technology, but it took the arrival of TigerGraph in 2017 to redefine what was possible with graph databases and bring critical maturity to this space.

The upcoming ISO GQL query language, which TigerGraph is helping to define, will benefit users by offering standard and expressive ways to build their applications. We believe GQL will become the new Python for data analytics, helping GQL and Graph Databases to take over the majority of advanced analytics currently poorly served by RDBMS and other in-house solutions.

Just as traditional BI tools (like Tableau, MS PowerBI, and Looker) democratized data analytics for business users who are not SQL experts, Graph-powered BI tools (like TigerGraph's Visual Query Builder) will democratize graph analytics, bringing the power of connecting and analyzing connected data to more first-time graph users.

Appendix

How easy is it for SQL developers to use a Graph database?

Recent industry interest in graph data has led to an accelerating rate of adoption of graph query development. The GQL standard's original motivation was to cater to data scientists regardless of their prior exposure to SQL. In contrast, TigerGraph sees enormous growth potential among the community of existing SQL developers, which is why our philosophy is to offer a smooth on-ramp from SQL to graph querying.

To that end, TigerGraph has designed a graph query language, GSQL, which extends SQL in the most natural and minimal way, exploiting the fact that both relational and graph queries can be seen as matching patterns against the data. GSQL serves as one of the reference languages informing the GQL standard, which has recently evolved to also include an SQL-flavored syntax inspired by GSQL.

How Graph powers Machine Learning and Artificial Intelligence

Today's analytical graph databases such as TigerGraph are taking organizations to another level by connecting all their data, representing knowledge better, and obtaining answers to deeper questions in real-time.

These benefits extend to the world of machine learning and AI. There are three ways in which graph databases and graph analytics can deliver smarter AI:

- Unsupervised learning with graph algorithms
- Feature extraction and enrichment with graph patterns
- In-database ML techniques for graphs

Learn all about it in this presentation.

How can you still be the leader if big RDBMS vendors or Cloud vendors or newer startups jump into the game?

We crush all of our competition at this time. Most of the time our customers choose TigerGraph because we are the only product that can meet their performance and scalability requirements. Our performance is often hundreds of times better than competitors.

How do we crush all of the competition? We built the native distributed graph database under stealth mode in the first five years from the ground up, using C++ to optimize performance, data compression, and memory usage for graph.

We are the only scalable distributed graph database in the market. Before TigerGraph, first-generation Neo4j was open source, leading the market, so there isn't much competition in the graph market. After TigerGraph launched out of stealth mode, there were some startups which tried to follow in our footsteps. They are still trying, but these startups are years behind us in terms of product maturity, performance, and scalability. Moreover, some are using design architectures that we considered and rejected, such as sparse matrix computation.

Today, our main competition on-prem is Neo4j, and the main competitor on Cloud is Amazon Neptune which is even slower than Neo4j. For a detailed comparison in performance, scalability, and functionality, please see our detailed Graph Database Buyer's guide for Neo4j and Amazon Neptune.

Appendix - Continued

Competitor analysis: Neo4j

Cons:

At a high level, Neo4j's biggest disadvantage is its single machine architecture which can only power a very limited number of use cases.

Pros:

Its biggest advantage is that it's the first mover in graph and has done marketing and community-building for 20 years to become well-known in this space. However, TigerGraph now has a strategic effort to build a developer community: a free cloud offering, a free on-prem enterprise edition, and numerous free online developer forums and events managed by our Developer Relations team. Coupled with our unparalleled performance and scalability, TigerGraph is now a recognized leader in the recent Forrester Wave report, getting a 5 out of 5 score on community and a total of nine criteria related to deployment at enterprise scale.

What about Neo4j 4.0 Fabric Architecture? We have analyzed Neo4j 4.0 Fabric architecture and compared it with TigerGraph for universally accepted enterprise scalability criteria for graph databases. You can find a detailed blog on the TigerGraph website.

Competitor analysis: Amazon Neptune

Cons:

The biggest disadvantage of Amazon Neptune (launched following the acquisition of a 10-person startup, Blazegraph a few years ago) is its single machine architecture and slower performance which is even worse than Neo4j's. At its root, Amazon Neptune is really an RDF triple-store graph that can be configured to work as a property graph, supporting a subset of the Gremlin/Tinkerpop language, an Apache open-source framework that is losing popularity. Neptune does not support OLAP analytical workloads.

Pros:

The biggest advantage of Amazon Neptune is that it's becoming well known because of the AWS brand. But on the plus side, this actually helps educate users and push more people to try Graph on the cloud, which is a big help to TigerGraph. After users become familiar with graph they'll easily find that only TigerGraph has the performance and scalability needed for those big projects, and we have many customers who have made that journey, testing Neptune but ultimately choosing TigerGraph.

The tech giants – Microsoft, Google, Facebook, IBM, and Oracle

Microsoft has Cosmos DB in the cloud which has very rudimentary graph APIs but is not a graph database. We don't view it as a competitor, rather we will seek to work more closely with Microsoft Azure to complement their graph offerings.

Appendix - Continued

Google has internal graph solutions for its graph algorithms such as PageRank but does not have a commercially available graph database that supports a high-level query language that is targeted for general purpose applications. As with Microsoft Azure, we will seek to work in partnership with Google GCP on graph.

Facebook supports Apache Giraph which has had no further development in the last few years, and it only supports offline graph analytics (no update, read-only, no query language, only Java API). Internally, Facebook has in-house graph solutions built on top of many MySQL systems for dedicated applications. These are narrow in scope and cannot be used for other applications.

Oracle supports graph on top of its RDBMS, but performance-wise it is not competitive to a native graph database like TigerGraph. In addition to performance, Oracle's graph is read-only – updates have to go through the RDBMS's tables.

IBM announced IBM Graph Cloud a few years ago based on JanusGraph. Due to the slow performance, we don't view the IBM graph as competition. Similar to Microsoft Azure, we'll seek to work with IBM Cloud in partnership.

TigerGraph's core advantages

Our design is closed-source (a trade secret), and our system is built from the ground up for parallel processing from graph storage to the graph processing engine.

TigerGraph's core advantages include:

- High performance due to native graph and ground-up design
- Scalable due to its distributed architecture
- Easy to use and powerful query language which is Turing-complete. This means the TigerGraph solution team and customers can often finish a Proof of Concept or pilot and demonstrate business value in days, not in weeks or months unlike using other systems.
- Graph visualization and BI. Because of the lack of a high-performance graph database in the past, there were no good graph visualization and BI tools. TigerGraph is the only engine to power real-time, interactive graph BI and visualization tasks. As such, TigerGraph brought innovation to the graph space via no code/low code tools like GraphStudio and Visual Query Builder.

Barriers our competition face

Building a distributed graph database requires time and substantial expertise. Take Neo4j as an example. After TigerGraph came out of stealth in late 2017, Neo4j tried to address its scalability problem as it had to respond to TigerGraph being the only distributed graph DB. The first approach Neo4j tried was to use the distributed capability in Spark to solve offline graph analytics, but this initiative failed.

Neo4j also tried to use more machines to solve the scalability problem, but instead of coming up with a real distributed graph system, Neo4j announced a federated approach called Neo4j Fabric which works by providing a single machine as a coordinator and scripts to talk to independent Neo4j machines, forcing developers to manually create graph schemas and queries on each independent machine (see details in the Buyer's Guide). Even though Neo4j has been working on graph for the past two decades, it is still difficult for them to build a distributed graph database.

Appendix - Continued

In addition to the technical challenge of building a distributed graph database, competitors face these additional barriers:

1. Even if they succeed in matching our capabilities, there won't be performance advantages over TigerGraph.
2. TigerGraph has been battle-tested by customers, and has added and continues to add enterprise features such as MultiGraph and various levels of data access control. When choosing a database, customers demand not only performance but also functionality, stability, security, and many other capabilities which also take years to build.
3. As an industry leader, TigerGraph is on the ISO GQL committee, leading the evolution of the standards. We bring new features to the committee, and we know in advance what is likely to be part of the GQL standard and its impact.
4. Because only TigerGraph has the performance and scalability needed by enterprises, big hardware vendors including Intel, HP Enterprise, Dell, and Xilinx are partnering with TigerGraph so that TigerGraph can run even faster on their hardware, meaning TigerGraph can continue to innovate in the high-performance graph space.
5. TigerGraph has a head start on no-code / low-code application development, graph BI, and graph visualization. We intend to further invest and bring more innovation to speed up the adoption of our graph database.
6. Graph+ML. We are adding more graph ML and AI capability to our system to make it easy and fast for data scientists to use a graph database. Again, because of our high performance, only TigerGraph can deliver the true benefits of Graph+ML.

About TigerGraph

TigerGraph is a platform for advanced analytics and machine learning on connected data. Based on the industry's first and only distributed native graph database, TigerGraph's proven technology supports advanced analytics and machine learning applications such as fraud detection, anti-money laundering (AML), entity resolution, customer 360, recommendations, knowledge graph, cybersecurity, supply chain, IoT, and network analysis. The company is headquartered in Redwood City, California, USA. Start free with tigergraph.com/cloud.