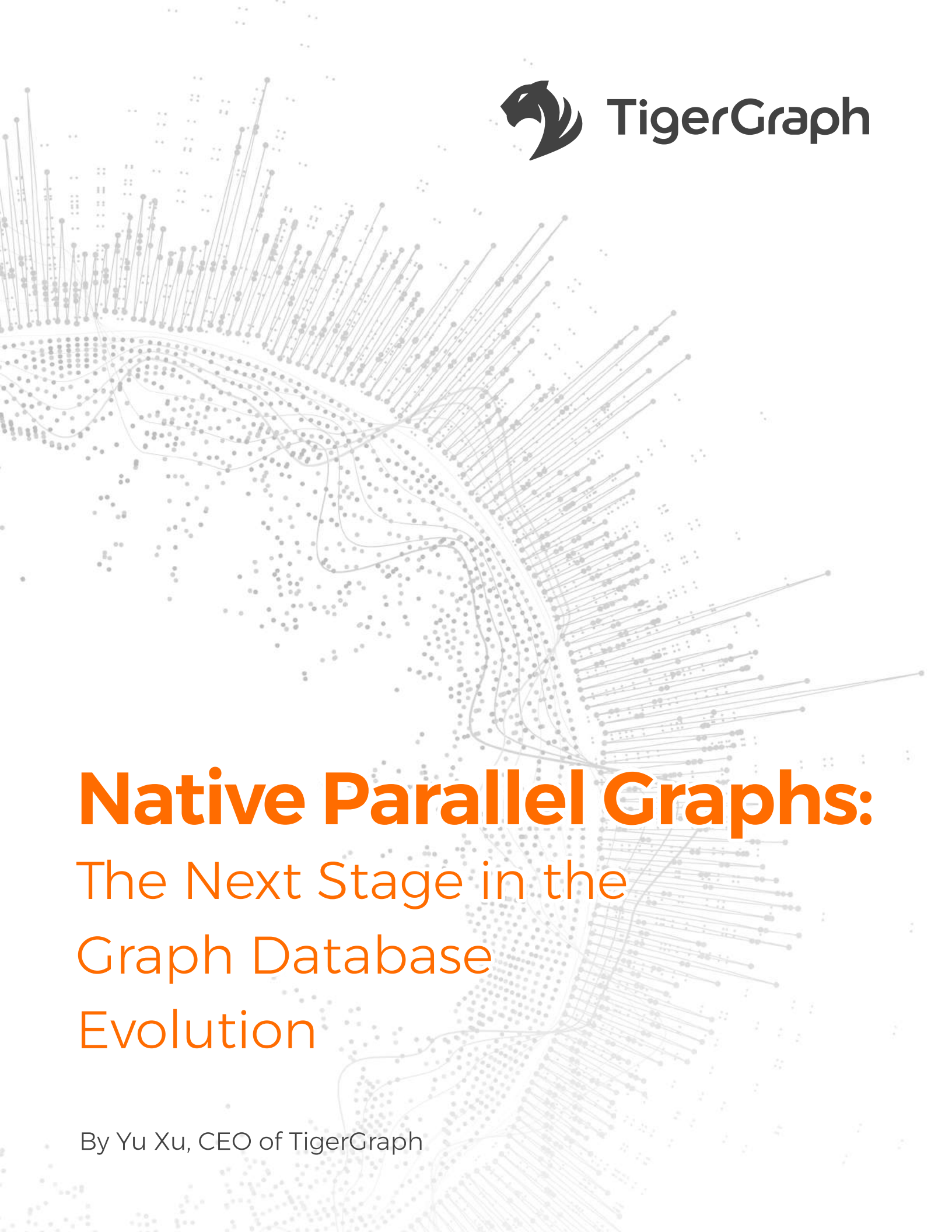


TigerGraph

Native Parallel Graphs:

The Next Stage in the Graph Database Evolution

By Yu Xu, CEO of TigerGraph

Graph databases comprise the fastest growing category of data management, according to DB Engines. They have gained quick adoption for several reasons. First, they overcome the challenge of storing massive, complex and interconnected data by storing data in a graph format, including nodes, edges and properties. They also offer advantages over both traditional RDBMS and newer big data products, including:

- Faster joining of related data objects
- Greater data set scalability
- More flexibility for evolving structures

While early generation graph technologies have provided benefits, they have been unable to support real-time analytics on an ever growing data ecosystem that continues to increase in both volume and complexity.

Introducing TigerGraph: The First and Only Native Parallel Graph

As the world's first and only Native Parallel Graph (NPG) system, TigerGraph is a complete, distributed, graph analytics platform supporting web-scale data analytics in real time. The TigerGraph NPG is built around both local storage and computation, supports real-time graph updates, and works like a parallel computation engine. These capabilities provide the following unique advantages:

- Fast data loading speed to build graphs - able to load 50 to 150 GB of data per hour, per machine
- Fast execution of parallel graph algorithms - able to traverse hundreds of million of vertices/edges per second per machine
- Real-time updates and inserts using REST - able to stream 2B+ daily events in real time to a graph with 100B+ vertices and 600B+ edges on a cluster of only 20

commodity machines

- Ability to unify real-time analytics with large scale offline data processing - the first and only such system

How Native Parallel Graph Differs from Early Generation Native Graph Technologies

There are three areas where early graph technologies and TigerGraph's Native Parallel Graph have significant differences.

1. Single Machine vs. Distributed System (Scalability) - Early generation native graph technologies cannot store a graph across multiple machines. Graphs are routinely larger than what a single machine can accommodate. With TigerGraph, the NPG scales out to multiple servers and has no graph size limitations.

2. Performance on a single server - Because of the built-in parallel computation capability, NPG can deliver 10x to 100x query performance improvements on a single machine. Meanwhile, the TigerGraph NPG is built to traverse 3 to 10+ hops in big graphs where early graph technologies simply time out.

3. Real time graph update speed - The TigerGraph NPG routinely delivers 10x loading speed thanks to its parallel computational capability.

TigerGraph NPG's Unique Design and Capabilities

TigerGraph's incredible loading speed, fast query execution, and real-time update capabilities are made possible by the following core platform technologies.

GRAPH STORAGE ENGINE & GRAPH PROCESSING ENGINE

Written in C++ for optimal performance, the TigerGraph NPG core system provides an integrated data technology stack that is both storage and compute focused. A native graph storage engine (GSE) works side-by-side with the graph processing engine (GPE) for fast and efficient processing of data and algorithms. The GPE is designed to provide built-in parallelism for a MapReduce-like computing model. The graph is stored both on disk and in-memory optimized for graph traversal and update, allowing the system to take advantage of the data locality on disk, in memory, and in CPU cache.

FAST DATA LOADING

TigerGraph's inherent parallelism enables it to load data 50 to 150 GB of data per hour, per machine. TigerGraph offers the most complete convenient loading environment, with bulk online loading, a built-in loading language, built-in indexing, and more.

HIGH DATA COMPRESSION

During data ingestion the TigerGraph system performs extremely efficient data compression and encoding. From 2x to 10x compression ratios (input data size to output graph size) are very common. Such compression reduces not only the memory footprint, but also cache misses, speeding up overall query performance. Additional performance

benefits are derived because TigerGraph operates directly on internally compressed and encoded data.

PARALLEL COMPUTATIONAL MODELS

Again, in the TigerGraph system, the graph is both a storage and a computational model. A vertex or an edge in the graph can store any amount of arbitrary information. Additionally, each vertex and edge can be associated with a compute function. Therefore, it acts as a parallel unit of storage and computation simultaneously. This is totally unique and unlike any existing graph database. With this approach, vertices and edges in the graph are not just static data units. Instead vertices are active computing elements that send messages to each other and respond via edges.

Unifying the MapReduce and parallel graph processing paradigms, TigerGraph's computing platform is based on the Bulk Synchronous Parallel (BSP) programming model which enables developers to implement a scalable parallel graph algorithm quickly and easily. A SQL-like graph query language (GSQL) provides for ad-hoc exploration and interactive analysis of Big Data. The TigerGraph system executes compute functions in parallel on every vertex/edge, taking advantage of multi-core CPU machines and in-memory computing. Enterprises can expect 100+ million vertex or edge traversals per second per server.

GRAPH PARTITIONING

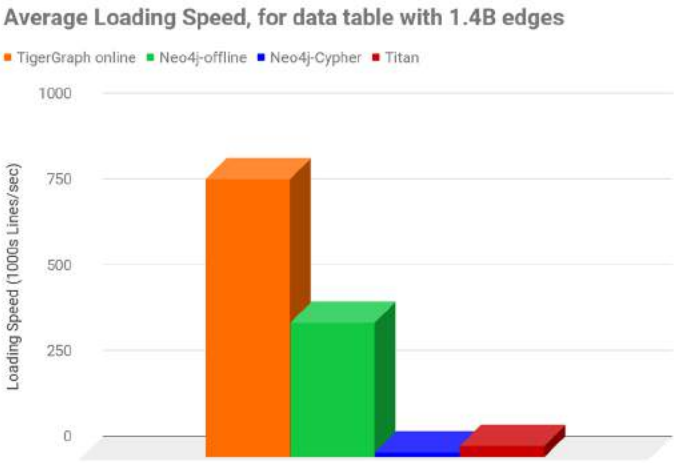
The TigerGraph system supports a variety of graph partitioning algorithms to deliver the best performance. In most cases, the partitioning is automatically performed on the input data and delivers great results without requiring optimization or tuning. The flexibility in the TigerGraph system allows application-specific and other mixed partitioning strategies to achieve even greater performance.

Built into the TigerGraph system is the ability to run multiple graph engines as an active-active network. Each graph engine can host identical graphs with different partitioning algorithms tailored for different types of application queries. The front-end server (typically a REST server) can route application queries to different graph engines based on the types of query.

REAL-TIME DEEP LINK ANALYTICS

Where current graph technologies are typically limited to two hops of traversal (i.e., two degrees of separation) within big graphs, TigerGraph's NPG enables 3 to 10+ hops of traversal. Each additional hop reveals additional connections and hidden insights. This deep link analytics includes fast graph traversal speeds of 100+ M vertex/edge traversals per second per server. It also includes fast data update of 100+ K updates per second per server.

SPEED AND SCALABILITY FOR VERY LARGE GRAPHS



TigerGraph supports parallel algorithms for very large graphs. Parallelism is employed even in single servers to both maximize speed and to ensure that performace scales as additional machines are added. This is incredibly important as graphs will invariably grow larger and larger in size. The graph storage engine has no upper limit for graph size; it will simply increase the address space and partition the graph across more machines. The Native Parallel Graph works for fast queries that touch anywhere from a small portion of the graph to millions of vertices and edges, as well as more complex analyses that must touch every single vertex in a billion-scale graph.

Summary

TigerGraph is the first and only Native Parallel Graph database system. Its fast, real-time technology empowers the exploration, discovery and prediction of complex relationships found hidden in the largest datasets. These capabilities are essential for critical enterprise applications, including personalized recommendations, fraud prevention, supply-chain logistics optimization, company knowledge base, and more.

About TigerGraph

TigerGraph is the world's first Real-Time Graph Analytics Platform powered by Native Parallel Graph (NPG) technology. TigerGraph fulfills the true promise and benefits of the graph platform by supporting real-time deep link analytics for enterprises with complex and colossal amounts of data. TigerGraph's proven technology is used by customers including Alipay, VISA, SoftBank, State Grid Corporation of China, Wish and Elementum.

Founded by Yu Xu, Ph.D. in 2012, TigerGraph is funded by Qiming VC, Baidu, Ant Financial, AME Cloud, Morado Ventures, Zod Nazem, Danhua Capital and DCVC. TigerGraph is based in Redwood City, CA. Learn more at www.tigergraph.com.