Graph Databases and Graph Analytics – Just a Hype or the End of the Relational World?

ITOUG Tech Day 2017

Hans Viehmann
Product Manager EMEA

Milano, June 8th, 2017

@SpatialHannes
Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Graph Data Model

• What is a graph?
  – Data model representing entities as vertices and relationships as edges
  – Optionally including attributes
  – Also known as „linked data“

• What are typical graphs?
  – Social Networks
    • LinkedIn, Facebook, Google+, Twitter, ...
  – Physical networks, Supplier networks,...
  – Knowledge Graphs
    • Apple SIRI, Google Knowledge Graph, ...
Graph Data Model

• Why are graphs popular?
  – Easy data modeling
    • „whiteboard friendly“
  – Flexible data model
    • No predefined schema, easily extensible
    • Particularly useful for sparse data
  – Insight from graphical representation
    • Intuitive visualization
  – **Enabling new kinds of analysis**
    • Overcoming some limitations in relational technology
    • Basis for Machine Learning (Neural Networks)
### Background: Three Types of Graph Data Models

<table>
<thead>
<tr>
<th>Property Graph Model</th>
<th>Social Network Analysis</th>
<th>RDF Data Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Graph Data Management</td>
<td>• Network path analysis</td>
<td>• Data federation</td>
</tr>
<tr>
<td>• Social Network Analysis</td>
<td>• Transportation modeling</td>
<td>• Knowledge representation</td>
</tr>
<tr>
<td>• Entity analytics</td>
<td></td>
<td>• Semantic Web</td>
</tr>
</tbody>
</table>

**Purpose**
- Property Graph Model: built for Linked Data and Semantic Web, conforming to W3C RDF standards
- Social Network Model: General Purpose Analysis
- Network Data Model: Purpose-built for Spatial Network Analysis
- RDF Data Model: Purpose-built for Linked Data and Semantic Web, conforming to W3C RDF standards
Categories of Graph Analysis

Computational Graph Analytics
- Compute values on vertices and edges
- Traversing graph or iterating over graph (usually repeatedly)
- Procedural logic
- Examples:
  - Shortest Path, PageRank, Weakly Connected Components, Centrality, ...

Graph Pattern Matching
- Based on description of pattern
- Find all matching sub-graphs

Categories: Person
- Person{100}
  - name = 'Amber'
  - age = 25
- Person{200}
  - name = 'Paul'
  - age = 30
- Person{300}
  - name = 'Heather'
  - age = 27

Categories: Company
- Company{777}
  - name = 'Oracle'
  - location = 'Redwood City'

Categories: Relation
- worksAt{1831}
  - startDate = '09/01/2015'
- friendOf{1173}
  - since = '08/01/2014'
- knows{2200}
- friendOf{2513}

More complex relationships and attributes can be represented and analyzed within this framework.
Examples for Graph Analytics

• Community detection and influencer analysis
  – Churn risk analysis/targeted marketing, HR Turnover analysis

• Product recommendation
  – Collaborative filtering, clustering

• Anomaly detection
  – Social Network Analysis (spam detection), fraud detection in healthcare

• Path analysis and reachability
  – Outage analysis in utilities networks, vulnerability analysis in IP networks, „Panama Papers“

• Pattern matching
  – Tax fraud detection, data extraction
Graph Analysis: Influencer Identification

• Requirement:
  – Identify entities from a graph dataset that are relatively more important than others (from topology)

• Approaches:
  – Determine centrality of entities (concept based on graph theory)
Graph Analysis: Influencer Identification

• Centrality is measure of relative importance of vertices in a graph
• Many variations of centrality in graph theory
  – Betweenness Centrality
  – Closeness Centrality
  – Eigenvector Centrality
  – Pagerank
  – HITS (Hyperlink-Induced Topic Search)
  – ...

(From Wikipedia)

Each algorithm suggests different definition of importance

(images from Wikipedia)
Graph Analysis: Influencer Identification

- Measuring importance using **Page Rank**
- Original algorithm developed by Larry Page for ranking in Google
- Making a node connected to by important nodes *also* important
- Can be measure of trust or prominence
Use case #1: Targeted Marketing in Telco

• Model each subscriber as a vertex in the graph
• Interactions between subscribers are represented by edges
  – Taking into account both on-net and off-net
• Based on call data records for voice, SMS, MMS
  – Usually combining all interactions in a property representing the strength of the edge
• Using centrality algorithms to determine important customers
• Target these customers with marketing campaigns for retention
  – Reducing churn risk for all additional customers he/she is connected with
Use case #2: Product Recommendation in Retail

• Requirement:
  – Need customer-item interactions such as purchases or rating records

• Approaches:
  – Create graph of customers and items
  – Run Personalized Pagerank using target customer as starting point
  – Optionally cluster customers for further analysis
  – (can also be used to find anomalies)
Graph Analysis: Anomaly Detection

• Requirement:
  – Identify entities from a large dataset that look different than others, especially in their relationships

• Approaches:
  – Define an anomaly pattern, find all instances of the pattern in the graph
  – Given nodes in the same category, find nodes that stand out (eg. low Pagerank value)
Use case #3: Fraud Detection in Healthcare

• Example for potential fraud detection
  – Public domain dataset
  – Medical providers and their operations

• Question
  – Are there any medical providers that are suspicious
    ➔ medical providers that perform different operations than their fellows
    (e.g. eye doctors doing plastic surgery)

• Approach
  – Create graph between doctors and operations
  – Apply personalized pagerank (a.k.a equivalent to random walking)
  – Identify doctors that are far from their fellows
Graph Analysis: Path and Reachability

• Requirement:
  – Identify all entities from a graph dataset that are connected with a given entity
  – Determine how entities are connected to each other (ie. via which paths)

• Approaches:
  – Traverse the graph starting from the specified vertex
Use case #4: Network Outage Analysis

• Real-world use case from a utilities company

• Analyzing power distribution network
  – Vertices: Generators, Transformers, Switches, ...
  – Edges: transmission lines

• Question
  – Which households have power when some given switches are turned off
Use case #4: Network Outage Analysis

• Represent the data as a graph
  – Fits very naturally
  – Note that vertices and edges have extra information or properties

• Answer the question in natural ways
  – Starting from the given vertex,
  – traverse the graph and mark reachable vertices
  – but without going through ‘off’ switches

Graph representation allows:
• Intuitive description of graph traversal
• Fast edge traversal without computing joins
• Network Intrusion Detection
  • Deep Learning + Graph Analysis

Property Graph
• Blue edges: malicious
• Other edges: normal traffic
• Many attacks originated from
  175.45.176.1 to target
  149.171.126.17
• Visualization tool: Cytoscape v3.2.1
  + Big Data Spatial and Graph v2.1
In-memory Analytics Engine – Product Packaging

Oracle Big Data Spatial and Graph

- Available for Big Data platform
  - Hadoop, HBase, Oracle NoSQL
- Supported both on BDA and commodity hardware
  - CDH and Hortonworks
- Database connectivity through Big Data Connectors or Big Data SQL
- Part of Big Data Cloud Service

Oracle Spatial and Graph (DB option)

- Available with Oracle 12.2 (EE)
- Using tables for graph persistence
- In-database graph analytics
  - Sparsification, shortest path, page rank, triangle counting, WCC, sub graph generation...
- SQL queries possible
  - Integration with Spatial, Text, Label Security, RDF Views, etc.
Oracle Graph Analytics Architecture

Graph Analytics
In-memory Analytic Engine

Graph Data Access Layer API
Blueprints & SolrCloud / Lucene

Scalable and Persistent Storage
Property Graph Support on
Apache HBase, Oracle NoSQL or Oracle 12.2

Java APIs

REST Web Service
Python, Perl, PHP, Ruby, Javascript, ...

Java APIs
The Property Graph Data Model

- **A set of vertices (or nodes)**
  - each vertex has a unique identifier.
  - each vertex has a set of in/out edges.
  - each vertex has a collection of **key-value** properties.

- **A set of edges (or links)**
  - each edge has a unique identifier.
  - each edge has a head/tail vertex.
  - each edge has a label denoting type of relationship between two vertices.
  - each edge has a collection of **key-value** properties.

https://github.com/tinkerpop/blueprints/wiki/Property-Graph-Model
Creating a Graph

• From a relational model
  – Rows in tables usually become vertices
  – Columns become properties on vertices
  – Relationships become edges
  – Join tables in n:m relations are transformed into relationships, columns become properties on edges

• Through API or interactively using a graphical tool
  – Adding vertices, edges, properties to a given graph

• From graph exchange formats
  – GraphML, GraphSON, GML (Graph Modeling Language)
Interacting with the Graph
No SQL and no SQL*Plus

• Access through APIs
  – Implementation of Apache Tinkerpop Blueprints APIs
  – Based on Java, REST plus SolR Cloud/Lucene support for text search

• Scripting
  – Groovy, Python, Javascript, ...
  – Apache Zeppelin integration, Javascript (Node.js) language binding

• Graphical UIs
  – Cytoscape, plug-in available for BDSG
  – Commercial Tools such as TomSawyer Perspectives
Graph Analysis Algorithms can be very hard to code ...

Oracle Big Data Spatial and Graph comes with 40+ pre-built algorithms

• Example: Find the size of the 2-hop network of vertices (Gremlin+Python)

```python
sum([v.query() \
    .direction(blueprints.Direction.OUT).count() \
    for v in OPGIterator(v0.query() \
    .direction(blueprints.Direction.OUT) \ 
    .vertices().iterator())])
```

• Single API call instead
  – Analysis in memory, in parallel

• Results can be persisted in Graph store and accessed from Oracle Database
  – Big Data SQL, Connectors
Example: Betweenness Centrality in Big Data Graph

Code

```
analyst.vertexBetweennessCentrality(pg) .getTopKValues(15)
```
Using Notebooks

Game Of Thrones

We have loaded the original data the representation of an undirected graph. Note that the number of edges is doubled due to how PGX internally represents an undirected graph.

One of the first measures of centrality - which characters are most important - mentioned in the article is eigenvector centrality.

Eigenvector centrality is an interesting measure of centrality because a vertex with few edges can be highly ranked if other highly ranked vertices are connected through it. So eigenvector centrality finds you characters that are the linchpins to the story even though they may not appear very often.

We will run the built-in algorithm for that, and visualize the results:

eigenvector = analyst.eigenvectorCentrality(undirected);
eigenvector.getTopKValues(10);

All fields:

| ID | value |

Keys

Groups

Values

value SUM

Tyrion
Sansa
Jaime
Catelyn
Cersei
Robb
Theon
Tywin
Arya
Robert
Catelyn

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.
Using Notebooks

Updated: Bitcoin Use Case - DBS

//create the graph
G = session.readGraphWithProperties("/opt/data/bitcoin/small_config_new.json")

PGX Graph named 'small_bitcoin_edges_new_2' bound to PGX session '4ed3dfe0-d5ed-4f79-bb05-a381822f7672' registered at PGX Server Instance running in embedded mode

Top nodes by degree centrality

has been a party to in the transaction network.

NodeID | Top_Degree
-------|----------
6       | 149,573  
537,797 | 50,487   
701     | 38,016   
14      | 32,646   
1       | 29,765   

Above, the left panel examines the top nodes by degree, and the right panel plots the distribution. The bitcoin network is a peer-to-peer platform, but a number of brick-and-mortar such as restaurants, apartments, law firms, and popular online services have been increasingly venturing into the space. Many of the highest degree nodes will belong to this category. On the other hand, the majority of nodes have very few connections. These are the individuals making isolated purchases.
Social Network Analysis Algorithms (1)

**Structure Evaluation**
- Conductance
- countTriangles
- inDegreeDistribution
- outDegreeDistribution
- partitionConductance
- partitionModularity
- sparsify
- K-Core computes

**Community Detection**
- communitiesLabelPropagation

**Ranking**
- closenessCentralityUnitLength
- degreeCentrality
- eigenvectorCentrality
- Hyperlink-Induced Topic Search (HITS)
- inDegreeCentrality
- nodeBetweennessCentrality
- outDegreeCentrality
- Pagerank, weighted Pagerank
- approximatePagerank
- personalizedPagerank
- randomWalkWithRestart
Social Network Analysis Algorithms (2)

Pathfinding
- fattestPath
- shortestPathBellmanFord
- shortestPathBellmanFordReverse
- shortestPathDijkstra
- shortestPathDijkstraBidirectional
- shortestPathFilteredDijkstra
- shortestPathFilteredDijkstraBidirectional
- shortestPathHopDist
- shortestPathHopDistReverse

Recommendation
- salsa
- personalizedSalsa
- whomToFollow

Classic - Connected Components
- sccKosaraju
- sccTarjan
- wcc
Pattern matching using PGQL

- SQL-like syntax but with graph pattern description and property access
  - Interactive (real-time) analysis
  - Supporting aggregates, comparison, such as max, min, order by, group by

- Finding a given pattern in graph
  - Fraud detection
  - Anomaly detection
  - Subgraph extraction
  - ...

- Proposed for standardization by Oracle
  - Specification available on-line
  - Open-sourced front-end (i.e. parser)

https://github.com/oracle/pgql-lang
PGQL Example query

- Find all instances of a given pattern/template in data graph
- Fast, scaleable query mechanism

```
SELECT v3.name, v3.age
FROM 'myGraph'
WHERE
  (v1:Person WITH name = 'Amber') -:FriendOf-> (v2:Person) -:Knows-> (v3:Person)
```

Query: Find all people who are known to friends of ‘Amber’.
OAAgraph integration with R

- OAAgraph integrates in-memory engine into ORE and ORAAH
- Adds powerful graph analytics and querying capabilities to existing analytical portfolio of ORE and ORAAH
- Built in algorithms of PGX available as R functions
- PGQL pattern matching
- Concept of “cursor” allows browsing of in-memory analytical results using R data structures (R data frame), allows further client-side processing in R
- Exporting data back to Database / Spark allows persistence of results and further processing using existing ORE and ORAAH analytical functions
Use case

- Load *persons* data into ORE
- Check the data set
- Cluster *persons* by their age with K-means

Load *calls* data into ORE
- Create an OAAGraph object with *persons* and *calls*
- Compute Pagerank and check results

- Export results back to ORE
- Cluster *persons* by their age AND pagerank values (with K-means)
Graph Analytics on SPARK

• Use SPARK for conventional tabular data processing (RDD, Dataframe, -set)

• Define graph view of the data
  – View it as node table and edge table

• Load into PGX

• Execute graph algorithms in PGX
  – Orders of magnitude faster than GraphX
  – More scaleable

• Push analysis results back into SPARK as additional tables

• Continue SPARK analysis

SPARK data structure and communication mechanism not optimized for graph analysis workloads
Text Search through Apache Lucene/Solr

- Use text indexing to access vertices or edges
  - Eg. find person with given name as starting point for reachability analysis
  - `oraclePropertyGraph.createKeyIndex("name", Vertex.class);`
  - `oraclePropertyGraph.getVertices("name", "*Obama*", true);`

- Based on Apache Solr/Solr Cloud
  - Highly scaleable through sharding and replication

- Uses Apache Lucene under the covers
  - open source text search engine library
  - inverted index, ranked searching, fuzzy matching ...

- Supports manual and auto indexing of Graph elements
In-memory Analytics Engine

Deployment options

**Interactive (private server) Execution**
- Client initiates PGX as a YARN task
- Client controls PGX via an interactive shell
- Example: `pgx> :loadGraph mygraph.json ...`
- Example: `pgx> :pagerank mygraph 0.85 ...

**Batch Mode**
- Client can submit a PGX script as a batch job
- `pgx> loadGraph ...

**Dry Run (Local Execution)**
- Client can run PGX locally with small data set
- Example: `...

**Shared Server**
- PGX can be configured as a service, with certain graphs pre-loaded
- And shared by multiple clients

**To load the Graph and run the analysis**
- Client
- PGX
- BDA Cluster
- HBASE / NoSQL

**Data File**
- Load data into PGX

**PGX Server**
- RDF / HBASE / NoSQL

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.
A Word on Performance

Sub-millisecond Performance for Graph Operations in NoSQL
Graph Analysis: Performance Compared with Neo4J

Path queries of Linux kernel source code

X86 Server
Xeon E5-2660 2.2Ghz
2 socket
x 8 cores
x 2HT
256GB DRAM

Neo4J: 2.2.1
Data:
- Linux kernel code as a graph
- Program analysis queries

Large performance advantage over Neo4J graph DB (2~4 orders of magnitude)
Distributed Graph Analysis Engine

Handling extremely large graphs

• Oracle Big Data Spatial and Graph uses very compact graph representation
  – Can fit graph with ~23bn edges into one BDA node

• Distributed implementation scales beyond this
  – Processing even larger graphs with several machines in a cluster (scale-out)
  – Interconnected through fast network (Ethernet or, ideally, Infiniband)

• Integrated with YARN for resource management
  – Same client interface, but not all APIs implemented yet

• Again, much faster than other implementations
  – Comprehensive performance comparison with GraphX, GraphLab
Graph visualization – Cytoscape, Vis.js, ...
Graph Visualization – Commercial Tools

Tom Sawyer Perspectives 7.5 has Property Graph pre-integrated
Linkurious Ogma

- Server-based (Node.JS)
- Light-weight JavaScript visualizer
- Powerful rendering
- Oracle integration

https://linkurio.us
https://www.slideshare.net/Linkurious/how-to-visualize-oracle-big-data-spatial-and-graph-with-ogma
https://linkurio.us/visualize-oracle-graph-data-ogma-library/
Summary

Graph capabilities in Oracle Big Data Spatial and Graph

• Graph databases are powerful tools, complementing relational databases
  – Especially strong for analysis of graph topology and multi-hop relationships

• Graph analytics offer new insight
  – Especially relationships, dependencies and behavioural patterns

• Oracle Big Data Spatial and Graph offers
  – Comprehensive analytics through various APIs, integration with relational database
  – Scaleable, parallel in-memory processing
  – Secure and scaleable graph storage on Hadoop using Oracle NoSQL or HBase

• Runs on commodity hardware or BDA, both on-premise or in the Cloud
Resources

• Oracle Big Data Spatial and Graph OTN product page:  
  www.oracle.com/technetwork/database/database-technologies/bigdata-spatialandgraph  
  – White papers, software downloads, documentation and videos

• Oracle Big Data Lite Virtual Machine - a free sandbox to get started:  

• Hands On Lab included in /opt/oracle/oracle-spatial-graph/  
  – Content also available on GITHUB under http://github.com/oracle/BigDataLite/

• Blog – examples, tips & tricks: blogs.oracle.com/bigdataspacialgraph

• @OracleBigData, @SpatialHannes, @agodfrin, @JeanIhm

• Oracle Spatial and Graph Group
Q&A
Integrated Cloud
Applications & Platform Services