From Single Instance to Horizontal Scalability

- Stability ↑↑↑
- Storage ↑↑↑
- Integration ↑↑↑
- Sharding +

Node To Tree To Graph
Evolution of Distributed Graph Systems

From Single Instance to Horizontal Scalability

- Standalone
- Master-Slave (i.e., HA, DR)
- Distributed Consensus (i.e., RAFT)
- HTAP
- Proxy (i.e., NameServer)
- Federation
- Shardling (i.e., Meta-server)
- ...
The Magic Quadrant of Graph DBMS

- Graph Computing Frameworks
- Non-Native Multi-model Graph Stores
- Some RDF Stores
- Some RDF Stores
- Some RDF Stores
- Some RDF Stores
- Some RDF Stores
- Non-Native Map-Reduce Graph Stores
- Native Graph Stores
- Some RDF Stores
- Some RDF Stores
- Small-Volume
- Volume
- Depth
- Shallow Computing
K-hop on Twitter-2010 Dataset
(42 Million Nodes & 1.47 Billion Edges with many supernodes)
Examine a graph's density

\[ \frac{|E|}{|V||(|V|-1)|} \rightarrow \left(2 \times \frac{|E|}{|V|}\right)^k \]

Task-1

```
root@twitter > khop().src({__id == 12}).depth(1:6).boost() as n
return count(n)
```

Total number of neighbors from 1-hop to 6-hop.
Total latency is 1.68 seconds
## 3 schools of Distributed Graph Systems

<table>
<thead>
<tr>
<th>Distributed Graph System</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
</table>
| Distributed Consensus with HTAP [3]       | • High Performance  
• Better ACID support  
• Small H/W footprint                                                  | • Vertical Scalability  
• Difficult to handle 10 billion plus nodes and edges                        |
| Proxy/Name-server/Grid or Federation      | • Balanced approach to scalability & performance  
• No data migration                                                     | • Non-transparent graph partitioning (human-logic based)                                      |
| Automated Shard                           | • Unlimited Scalability  
• Great meta-data query and ingestion performance  
• Sophisticated Cluster Management                                      | • Degraded graph query performance  
• Sophisticated Cluster Management  
• Large H/W footprint                                                      |
## Scenarios of Distributed Graph Systems

<table>
<thead>
<tr>
<th>Type</th>
<th>Characteristics</th>
<th>Business Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Density Parallel Graph Computing (HDPC)</td>
<td>• Real-time read/write data, online processing &amp; calculation&lt;br&gt;• Ideal for deep range queries</td>
<td>• Transaction interception&lt;br&gt;• Online Anti-fraud&lt;br&gt;• Anomaly detection&lt;br&gt;• Real-time recommendation&lt;br&gt;• AI/ML Augmentation&lt;br&gt;• Other real-time scenarios</td>
</tr>
<tr>
<td>HDPC &amp; Shard</td>
<td>• Separation of read/write operations&lt;br&gt;• Elastic compute nodes [19] for shard/offline data</td>
<td>• Knowledge Graph&lt;br&gt;• LLM Augmentation&lt;br&gt;• Indicator calculation&lt;br&gt;• Audit&lt;br&gt;• Cloud Data Center&lt;br&gt;• Graph at the core of IT Infra.</td>
</tr>
<tr>
<td>Shard</td>
<td>• Meta-data oriented&lt;br&gt;• Shallow neighborhood calculation (1-2 hop) only</td>
<td>• Archive&lt;br&gt;• Data Warehouse&lt;br&gt;• Data Science</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td><strong>Peer</strong></td>
<td><strong>127.0.0.1:40061</strong></td>
<td><strong>HDPC / Computing &amp; Storage Server/Instance</strong></td>
</tr>
<tr>
<td><strong>Shard</strong></td>
<td>[Peer1…3]</td>
<td><strong>HTAP Cluster</strong></td>
</tr>
<tr>
<td><strong>NameServer</strong></td>
<td>[Shard1, Shard2]</td>
<td><strong>Management &amp; Computing Server</strong></td>
</tr>
<tr>
<td><strong>NameServer Cluster</strong></td>
<td>[N1,N2]</td>
<td><strong>Multiple NameServer</strong></td>
</tr>
<tr>
<td><strong>Elastic Compute Node</strong></td>
<td>[Peer1, Peer2]</td>
<td><strong>Dynamically allocate compute nodes</strong></td>
</tr>
<tr>
<td><strong>Meta</strong></td>
<td>[127.0.0.1:50061]</td>
<td><strong>Configuration&amp;Listener</strong></td>
</tr>
</tbody>
</table>
find().nodes() as nodes => [1,2,3,4]
khop().src().depth(3)
uncollect [“1”, “2”, “3”, “4”] as nodes
khop().src({_id == nodes}).depth(3)

Khop(1) on CN1
Khop(2) on CN2
Khop(3) on CN3
Khop(4) on CN1
A-to-B Path Parallelization

Dynamic CN

Chains/Broadcast

find().nodes({_id == 1}) as src
find().nodes({_id == 2}) as dest
ab().src(src).dest(dest).depth(1:2)
Cluster Hierarchy (as a Graph)

Meta

NameServer Cluster

NameServer

Elastic Compute Node

Shard[Peer]
Distributed Ultipa 5.0 Summary

• Type 1: Data are processed on name servers (or proxies)
• Type 2: Data are processed on shard servers and name servers (Peer-to-peer architecture).
• Exchange Operator between Relational Data Stream(s) and Graph Algo
• Optimization of relational data flow (as start node/edge) is necessary in restricted range graph queries.
HTAP/Instance vs. Shard

Latency (in seconds)

- Online Data Ingestion
  - HTAP 3-instance: 2736
  - HTAP 3-shard: 10800
- K-Hop (K=3)
  - HTAP 3-instance: 0.7
  - HTAP 3-shard: 11
- Shortest Path
  - HTAP 3-instance: 1.8
  - HTAP 3-shard: 28
- PageRank
  - HTAP 3-instance: 16
  - HTAP 3-shard: 161
- LPA
  - HTAP 3-instance: 101
# Three schools of Distributed Graph Systems

<table>
<thead>
<tr>
<th>Distributed Graph System</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed Consensus with HTAP [3]</td>
<td>• High Performance</td>
<td>• Vertical Scalability</td>
</tr>
<tr>
<td></td>
<td>• Better ACID support</td>
<td>• Difficult to handle 10 billion plus nodes and edges</td>
</tr>
<tr>
<td></td>
<td>• Small H/W footprint</td>
<td></td>
</tr>
<tr>
<td>Proxy/Name-server/Grid or Federation</td>
<td>• Balanced approach to scalability &amp; performance</td>
<td>• Non-transparent graph partitioning (human-logic based)</td>
</tr>
<tr>
<td></td>
<td>• No data migration</td>
<td></td>
</tr>
<tr>
<td>Automated Shard</td>
<td>• Unlimited Scalability</td>
<td>• Degraded graph query performance</td>
</tr>
<tr>
<td></td>
<td>• Great meta-data query and ingestion performance</td>
<td>• Sophisticated Cluster Management</td>
</tr>
<tr>
<td></td>
<td>• Sophisticated Cluster Management</td>
<td>• Large H/W footprint</td>
</tr>
</tbody>
</table>
Ultipa builds the world’s fastest graph database and killer apps that empower smart enterprises with graph-augmented business data intelligence.

**Key Features of Ultipa Graph DBMS:**
- Micro-second Query Processing & Ultra-Deep Data Penetration
- Real-time Attribution/Contribution Analysis
- Real-time Stress Testing & Scenario Simulation
- Highly Visualized 3D Interactive Web GUI
- HTAP-MPP Cluster and Fast Deployment & Migration

**Ultipa Product Matrix:**
- Real-time Decision Making & Anti-Fraud System
- Intraday Liquidity Risk & Cash Management System
- Real-time Asset & Liability Management System
- Smart Data Intelligence Toolkits
- Smart BI & Advanced Analytics

**Contact Us**
www.ulptipa.com
support@ulptipa.com

---

**Data Governance**
Graph to Relate Everything
Penetrate Everything
Quantify Everything

**Agility & Capability**
High-dimensional Correlation Analysis w/ Finest Granularity
Real-time RCA (Root-Cause)

**Computing Power**
Faster by 10,000x
100+ Algorithms
30-Hop Plus

**Killer-App**
Data Intelligence Toolkits
Asset-Liability Management
Smart BI/RTD Applications

Contact: [ricky@ulptipa.com](mailto:ricky@ulptipa.com)
+1-408-917-0675
[https://www.linkedin.com/in/rickysun](https://www.linkedin.com/in/rickysun)

Web: [https://www.ulptipa.com](https://www.ulptipa.com)
Ultpa V4
HTAP Distributed Consensus

Success Stories:
Deployed with G-SIB banks, stock exchanges and insurance companies. Largest commercial deployment of 100B+ graph size.
Ultipa V5
Horizontal Scalability
Low-mem Consumption
(GA in 2023)

Sophisticated scalable graph database system w/ unlimited scalability, deep-data processing, and elastic computing capabilities.