Introduction to Ultipa Graph

GUG 2nd Seminar, 22023.09.23

Agenda

01. Intro

02. Ultipa Graph

03. Drivers, UQL and algorithms

04. Ultipa & South Korea GUG

01 Intro, Yuri Simione

Yuri Simione

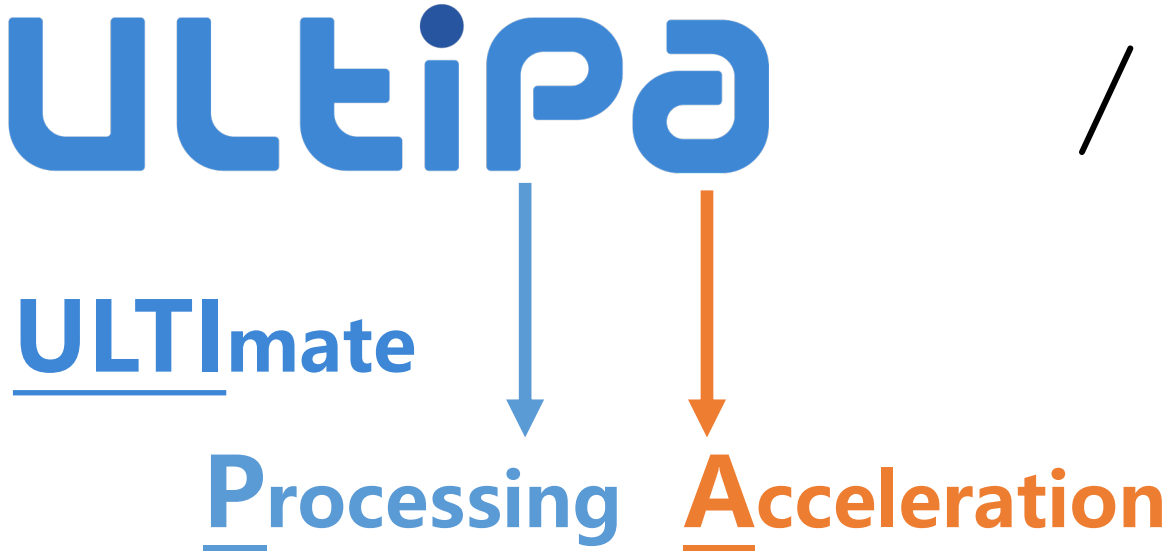
Sales Director

Ultipa, Inc.

Master of Science (MSc), Computer Science, University of Pisa. Yuri has devoted a significant portion of his career to unstructured information management. He worked as a consultant for companies like Xerox and EMC (now Dell EMC), specializing in products such as Opentext Documentum and Adobe AEM. Since 2014, he has shifted his focus to the world of graph databases and graph analytics. As of 2022, he serves as the Sales Manager for Ultipa, the California-based startup behind the eponymous graph database, which is the most performant and innovative in the market.

LinkedIn: https://www.linkedin.com/in/yurisimione/
Twitter: https://twitter.com/artika4biz
Email yuri.simione@ultipa.com
Mob: +39 333 4156248

01 Intro, Ultipa, Inc.



LinkedIn: https://www.linkedin.com/company/ultipa/
Twitter: @Ultipa_Graph
Web: https://www.ultipa.com/

Intro, Ultipa, Inc.

Few years ago, the current founders of Ultipa's needed a graph database that was:

- Very fast
- Scalable
- Capable of **multi-hop** graph traversal (+30 hops)

They tried several graph databases, but none met the performance, low latency, and deep multi-hop graph traversal requirements.

So, they decided to create a completely new graph database and a new company: Ultipa!

01 Intro, Ultipa, Inc.

- Founded in 2019
- Headquarters in San Ramon, California
- Offices in China, Hong Kong, Singapore, Italy
- More than 100 employees
- Secured $21M in Series-A financing
- Vision: Building **next-generation** leading **graph database products**, and *empowering smart enterprise w/ graph augmented intelligence.*

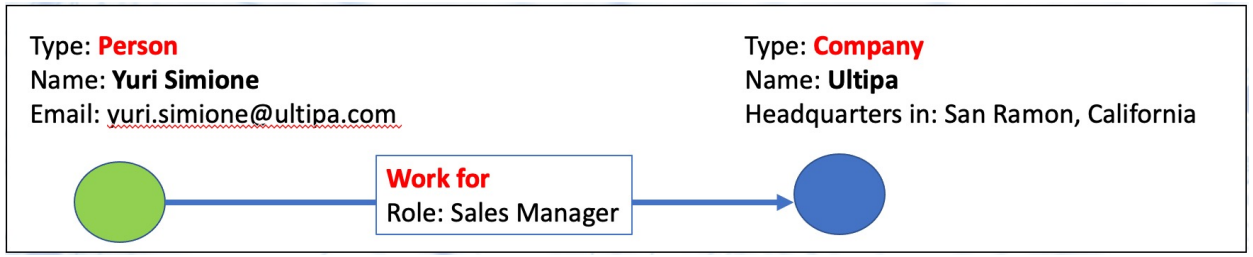
02

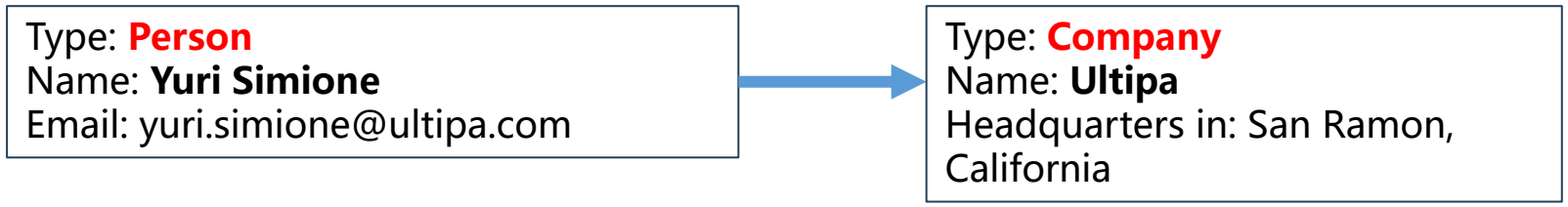
Ultipa Graph

02 Ultipa is a Native Graph Database

Graph Processing (Query, algorithms)

Graph Storage

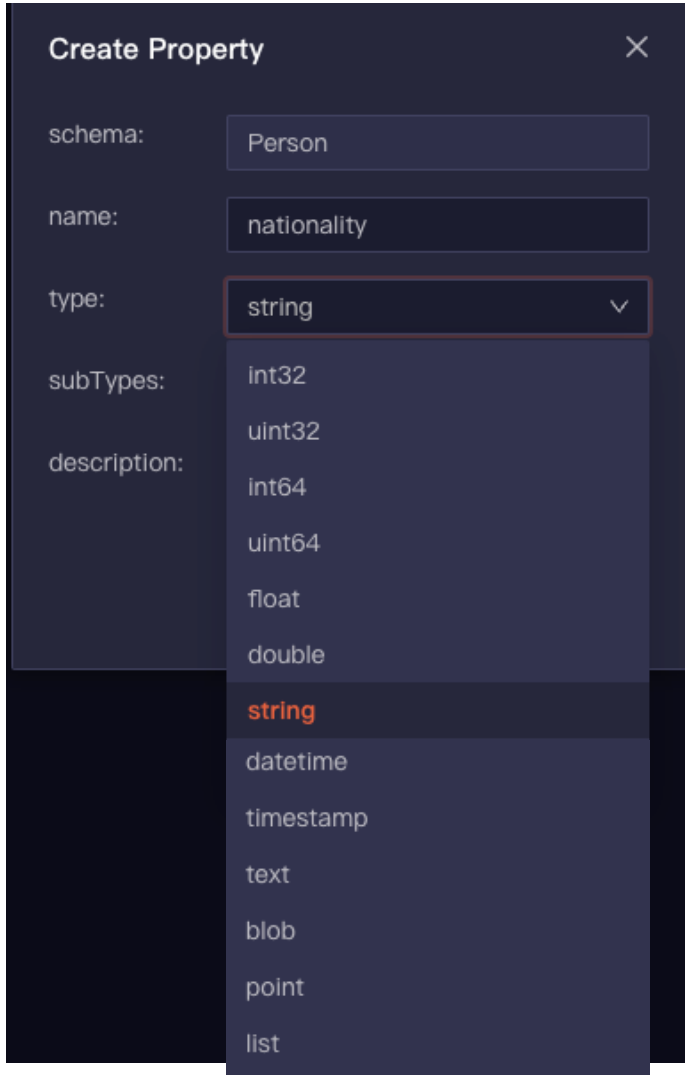


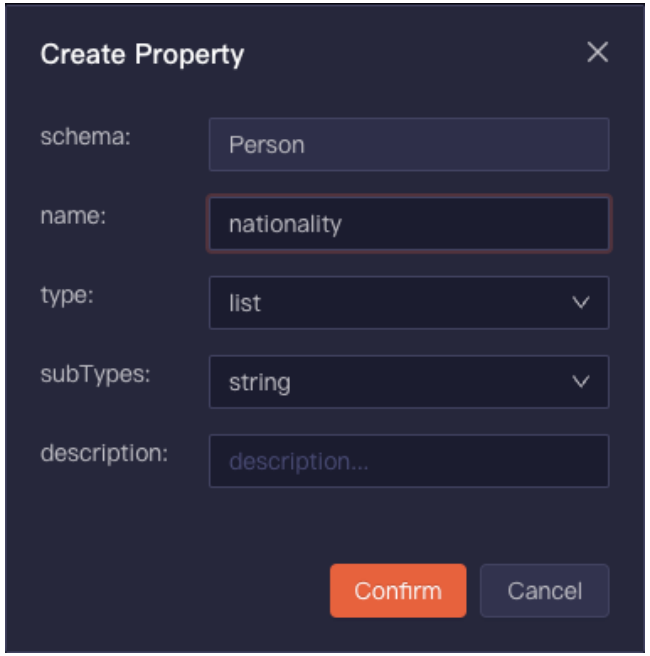


Index-free adjacency: Accessing nodes and relationships in a native graph database is an efficient, constant-time operation and allows you to quickly traverse millions of connections per second

02

Ultipa is a Property Graph Database





Schemaless VS schemafull

A **schemaless graph** database is a type of database system designed to store and manage graph data without a predefined schema or structure:

- Users can add nodes and edges to the database **without specifying a fixed schema in advance**.
- Users can attach properties of various data types to nodes and edges **without predefined rules**.
- Two instances of the same node or edge type can have **different set or properties.**
- Offer the freedom to query and traverse the graph without being constrained by a predefined schema.
- **Two developers can use different property names to store the same data (ID, SSN, PERSONAL_ID, personalID)**

A **schemafull graph** database is a type of database system designed to store and manage graph data with a predefined schema or structure:

- Users must adhere to this predefined schema when adding data to the database.
- Schemafull graph databases enforce strict data type constraints .
- Due to the fixed schema, querying and traversing the graph follows a predefined structure. Queries are typically more predictable and constrained compared to schemaless databases
- The schema plays a crucial role in maintaining data consistency.

Ultipa Graph is demi-schema

Like a schemafull database, Ultipa Graph requires the definition of a schema in advance

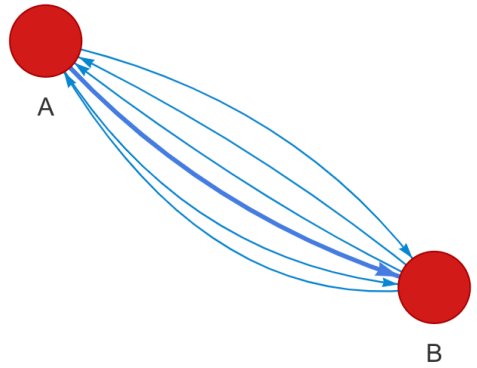
Like a schemaless database, Ultipa Graph offers the freedom to query and traverse the graph without specifying a predefined schema.

A Product and a Service are similar concepts but can be described by two different sets of properties. In Ultipa Graph, we can model these concepts defining two node types, PRODUCT, SERVICE. Both can have a property named **price**:

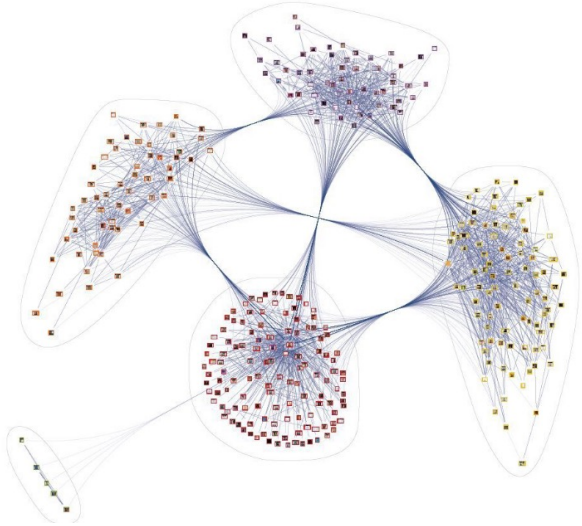
```
find().nodes({@PRODUCT && price <= 10} as myproduct return myproduct
find().nodes({@SERVICE && price <= 10} as myservice return myproduct

find().nodes({price <= 10}) as myselection return myselection
```

Demi-Schema is the best of both worlds!

02 Multigraph and supernodes



Unlike simple graphs, where each pair of nodes is connected by at most one edge, **multigraphs** allow multiple edges between the same pair of nodes. These multiple edges are often referred to as "parallel edges."



Supernodes are graph nodes that are directly connected to a massive number of other nodes.

Ultipa has the capacity and the performance to allow you to include **multigraphs** and **supernodes** in your analytic

Ultipa is a translytical graph database

Ultipa is a translytical* graph database as it combines the capabilities of both transactional (OLTP) and analytical (OLAP) databases into a single system. It is designed to handle both transactional processing and graph analytical processing.

1.Real-Time Analytics: Ultipa can perform complex analytical queries and execute graph algorithms on live transactional data in real-time or near-real-time, allowing organizations to make data-driven decisions without the need for separate processes, like those used by other graph databases.
2.High Performance: Ultipa is optimized for both read and write operations, ensuring fast response times for transactional tasks as well as graph analytical queries, even with large datasets.
3.Scalability: Ultipa is designed to scale horizontally and vertically to accommodate growing data volumes and user loads, ensuring that performance remains consistent as data and workload increase. Ultipa can manage hundred of billions of nodes/edges.

(*) Translytical is also referred as HTAP, Hybrid Transactional Analytical Processing

Ultipa Graph is incredible fast

- Ultipa has been designed and developed with shared-nothing architecture that's capable of HTAP, MPP and high-availability.
- Every component is written in C++
- For example, Ultipa provides full text indexing/search. The fulltext engine is also written in C++ and integrated in the graph engine, it is not an external component
- Graph algorithms are written in C++

Ultipa Patents

- Mass Volume Data Import Method, Device and Storage Medium (202010312564 4)

- Graph Data Processing Method, Device, and Storage Medium (202010331377 0)

- Graph Database Deep Path Search and Dynamic Pruning Method (202010817043 4)

- Graph Database's Super Node Oriented Data Processing Method, Device and Storage Medium (202011082674 2)

- A Flexible Graph Data Schema Design Method — Demi-schema (202111640262.0)

- An HTAP Capable Graph Data Query Processing Method (202111640279.6)

We have many trade secrets that we cannot publicly disclose

Benchmark

https://www.ultipa.com/article/benchmarks/ultipa-graph-benchmark-report

Performance Benchmark

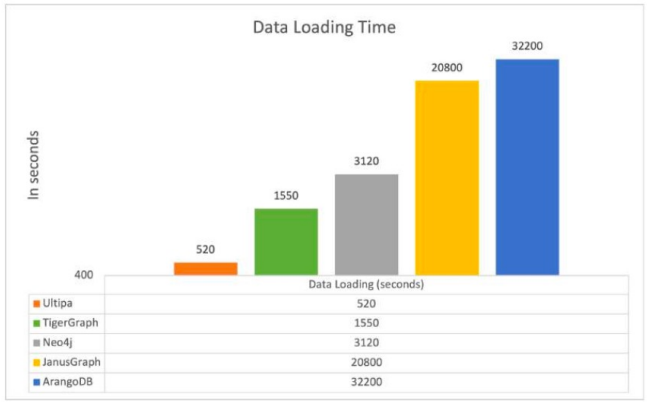
Ultipa has recently released v4.0 of its flagship graph database product, on top of its v3.0's **already world-leading performance**. Ultipa 4.0 introduced performance improvements of 50% with a 30% reduction in memory usage

In a 2022 benchmark test using Twitter-2010 dataset (42M vertices and 1.47B edges), Ultipa shows[1] great performance advantage over other graph systems (Neo4j, Tigergraph, and ArangoDB) using a typical 3-instance PC-server cluster. Very consistent benchmarking/stress-testing results have been achieved using any kind of commercial or academic/public datasets:

- **Data Ingestion**: Ultipa is 3-60 times faster than any other graph database.
- **K-Hop or Shortest Path**: Ultipa is 10-1,000 times faster than any other graph database. For ultra-deep (>6 hops) queries, Ultipa is the only system that can return with correct results.
- **Graph Algorithms** (PageRank, LPA, Louvain, Jaccard Similarity, Random Walk, etc.): Ultipa is at least 10 times faster while some systems couldn't return at all.

LOADING TEST

Loading the entire dataset into graph database and start providing services. This test can show how fast a graph system ingest large-volume of data.



Data Loading (seconds)	
Ultipa	520
TigerGraph	1550
Neo4j	3120
JanusGraph	20800
ArangoDB	32200

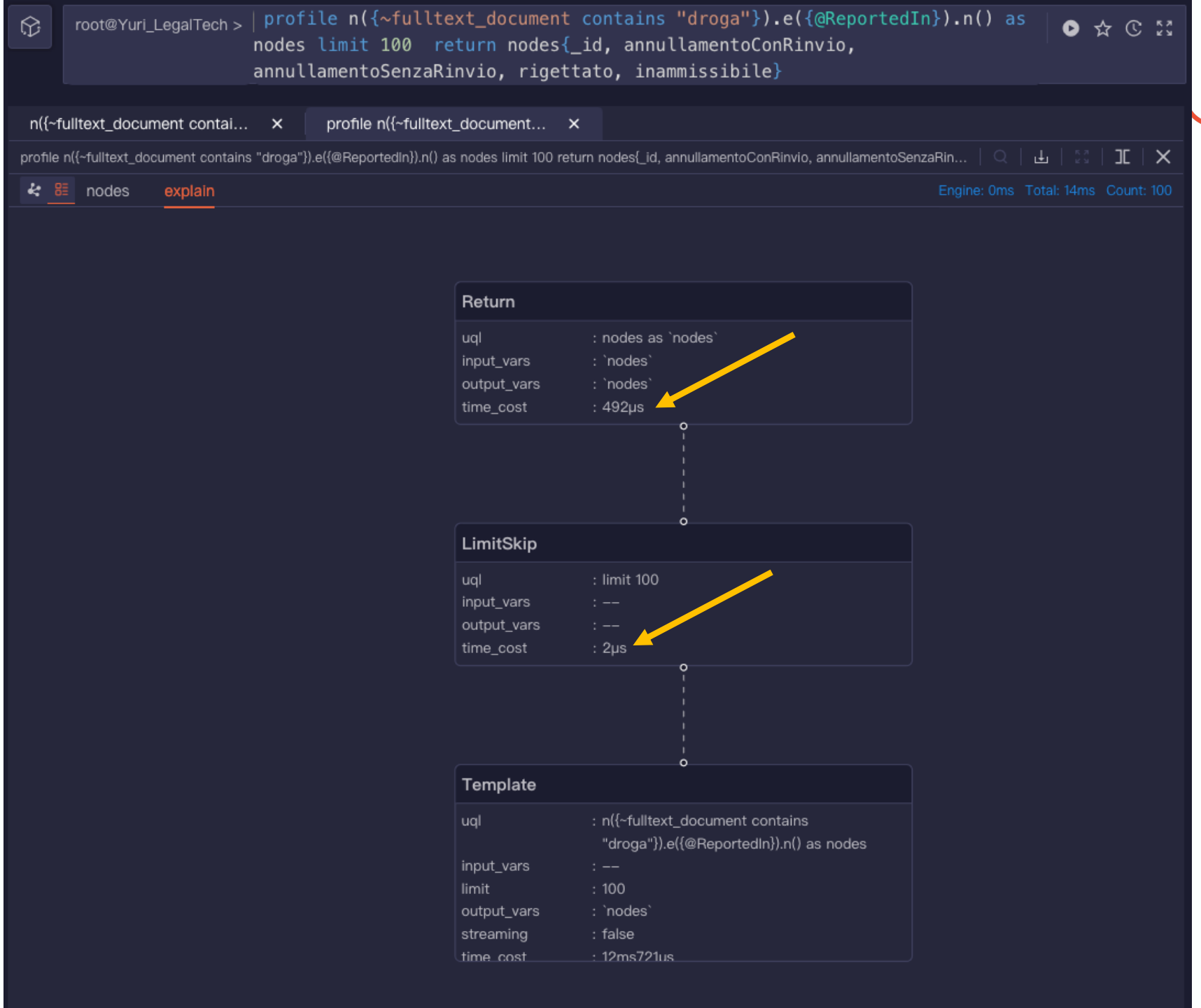
K-HOP NEIGHBOR QUERYING

K-hop query is a fundamental type of query with any graph system. It allows user to quickly identify the scope of impact of an entity (the subject vertex). K-hop must be implemented using BFS (Breadth-First-Search) method, and there are a few caveats:

- There are usually 2 ways of defining K-Hop and the results are different. One way is all neighbors from the 1st hop all the way to the Kth hop, the other way is the neighbors exactly Kth-Hop away from the starting vertex.
- If a vertex appears on the Nth Hop of a specific vertex, it will NOT appear on any other hop. While executing the algorithm, it's pivotal to conduct de-duplication of vertices across different hops, otherwise, the results would be wrong.

	1-Hop	2-Hop	3-Hop	6-Hop
Ultipa	0.00062	0.027	0.52	1.408
TigerGraph	0.024	0.46	6.6	62.5
Neo4j	0.2	18.3	290	
JanusGraph	0.39	27.7	4300	
ArangoDB	1.667	28.9	3888	

Ultipa Graph is so fast that the microsecond becomes the metric for assessing performance!



02

UltipaGraph

is a native, translytical, property graph database, that can handle multipgraphs, supernodes. It is demi-schema and it can handle hundred of billions of nodes/edges. It is capable of +30 hops queries. It is super-fast.

UQL, Ultipa Query Language

- It is the language exclusive for Ultipa Graph Database and Graph Computing Engine.
- DQL (Data Query Language): to query nodes, edges and paths in the graph;
- DDL (Data Definition Language): to add/delete GraphSet, modify schema, define properties, create index etc.;
- DML (Data Manipulation Language): to insert, delete and update the metadata and other content of GraphSet;
- DCL (Data Control Language): mainly used to manipulate database permission settings, such as user management, role management, grant and revoke of permissions etc.

https://www.ultipa.com/document/ultipa-graph-query-language/k-hop/v4.3

Comparison of GQLs: Cypher, GSQL & UQL

label: Person
Properties: {
name: "Areith",
age: 18
}
Schema: Person

{relation: "is" }

{Schema: "jobis" }

label: Job
Properties: {
name: "Chef"
}
Schema: Job

```
Match path = (p:Person) – [{relation:"is"}]-(j:Job)
Where p.name = "Areith" && j.name== "Chef"
return path
```

Cypher (label-only)

```
g.V().hasLabel('person').has('name',"areith").outE().ha
sLabel("jobIs").V().hasLabel("job").has("name","chef
.path()
```

Gremlin

Confusing

```
n({ @person.name == ”Areith" }) . e({ @jobis }) .n({
@job.name == ”Chef " })
as paths
return paths
```

UQL Schema

```
n({ name == ”Areith" }) . e() .n({ name == ”Chef " })
as paths
return paths
```

UQL Schema-Free

Demi-Schema

```
CREATE QUERY areithjob(vertex<word> w) for graph
test {
  SetAccum<node> @@nodeSet;
  SetAccum<edge> @@edgeSet;

  Start = {persion.*};
  Result = select j from Start::p - (jobIs:e) - job:j
           WHERE  p.name == "areith" AND j.name ==
"chef"
        accum @@nodeSet += p,
        accum @@nodeSet += j,
        accum @@edgeSet + = e;

  print @@nodeSet;
  print @@edgeSet;
}
```

GSQL

Too Complex

Online Document: https://www.ultipa.com/article/technical/graph-query-languages-simplicity-speed

UQL, Ultipa Query Language

- We do not support openCypher, but we provide a tool to migrate Cypher queries to UQL

```
Cypher -->  MATCH (n) RETURN n
UQL ==> ✅  find().nodes({}) as n RETURN n
Cypher -->  MATCH (movie:Movie) RETURN movie.title
UQL ==> ✅  find().nodes({@Movie}) as movie RETURN movie.title
Cypher -->  MATCH (director {name: 'Oliver Stone'})--(movie) RETURN movie.title
UQL ==> ✅  n({name == 'Oliver Stone'} as director) .e({} ). n({} as movie) RETURN movie.title
Cypher -->  MATCH (:Person {name: 'Oliver Stone'})--(movie:Movie) RETURN movie.title
UQL ==> ✅  n({@Person && name == 'Oliver Stone'} ) .e({} ). n({@Movie} as movie) RETURN movie.title
Cypher -->  MATCH (:Person {name: 'Oliver Stone'})-->(movie) RETURN movie.title
UQL ==> ✅  n({@Person && name == 'Oliver Stone'} ) .re({} ). n({} as movie) RETURN movie.title
```

- We will support GQL, when this ISO standard will be released (2024)

https://www.ultipa.com/document/ultipa-graph-query-language/k-hop/v4.3

03

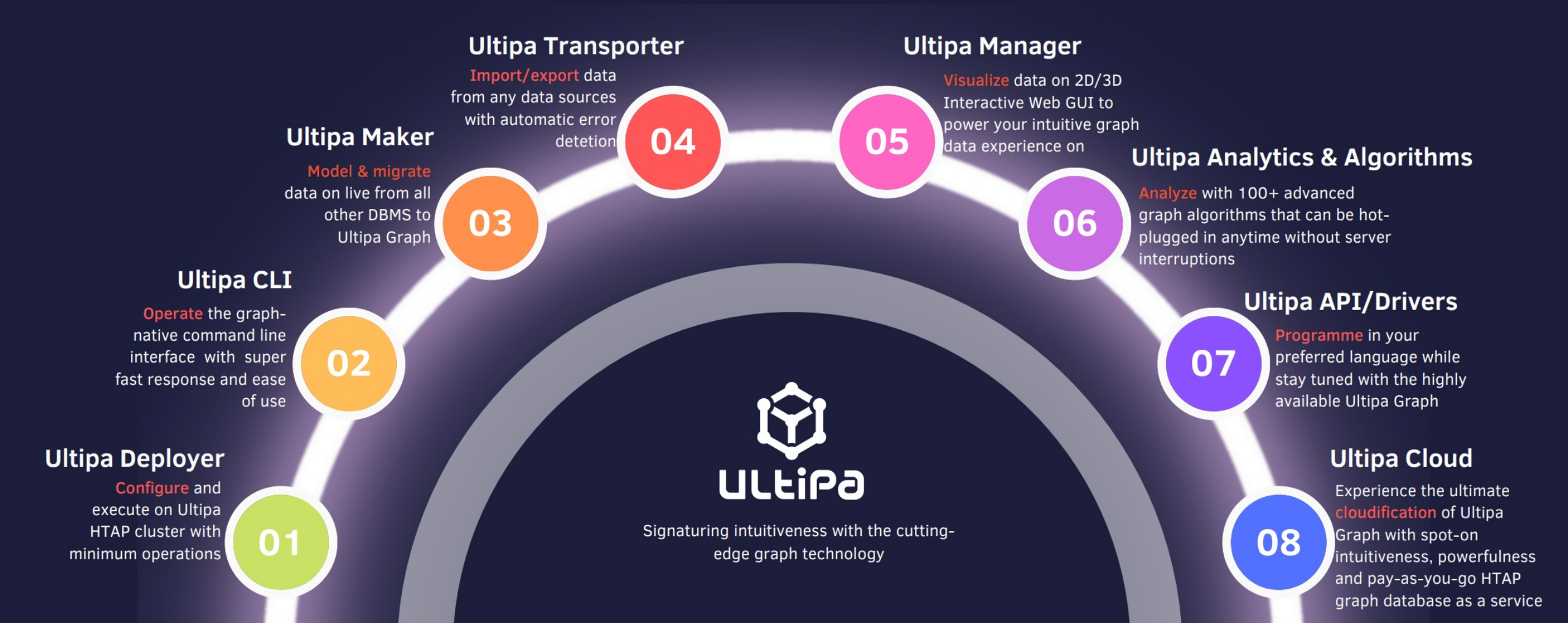
Ultipa Drivers

A set of native SDKs to build custom applications. We support:

- Restful API
- Java
- Python
- Go
- NodeJS

Ultipa Toolchain

With the world's only 4th-generation graph database, Ultipa offers an unprecendented graph computing experienece built on an innovative and intuitive graph toolchain.



03

Library of Graph Algorithms

- Hot-pluggable, hot-updatable
- Written in C++
- More than 50 (actually more than 100!)
- 9 graph embedding algorithms
- We provide SDK to create new algorithms

Graph Algorithms

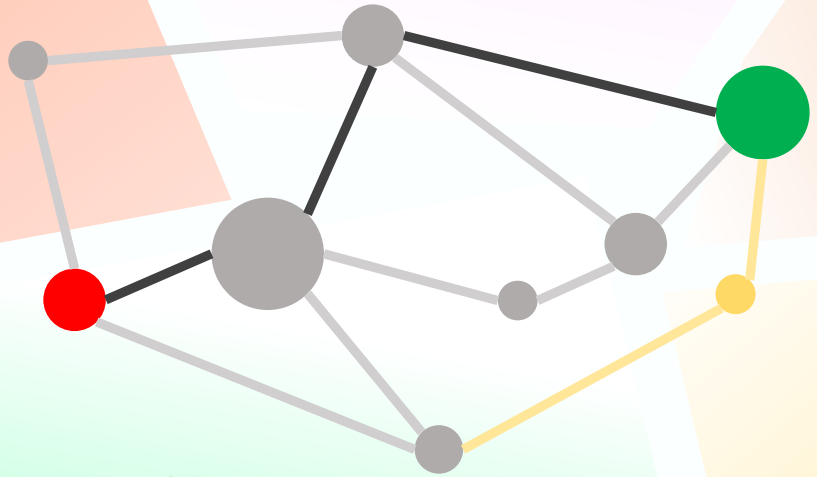
Graph Accelerated Data Science

Graph Embedding

- Random Walk
- Node2Vec
- Struc2Vec
- LINE
- Fast RP
- GraphSage

Degree & Similarity

- Degree
- Centrality
- Closeness
- Jaccard Similarity
- Cosine Similarity
- Adamic-Adar



Community Detection

- Page Rank
- Sybil Rank
- Label Propagation
- HANP
- Louvain

General Algorithms:

- BFS/DFS
- K-Hop-All
- Connected Component
- Triangle Counting
- Circle-Detection
- Common Neighbors
- Induced Subgraph
- Bipartite Graph

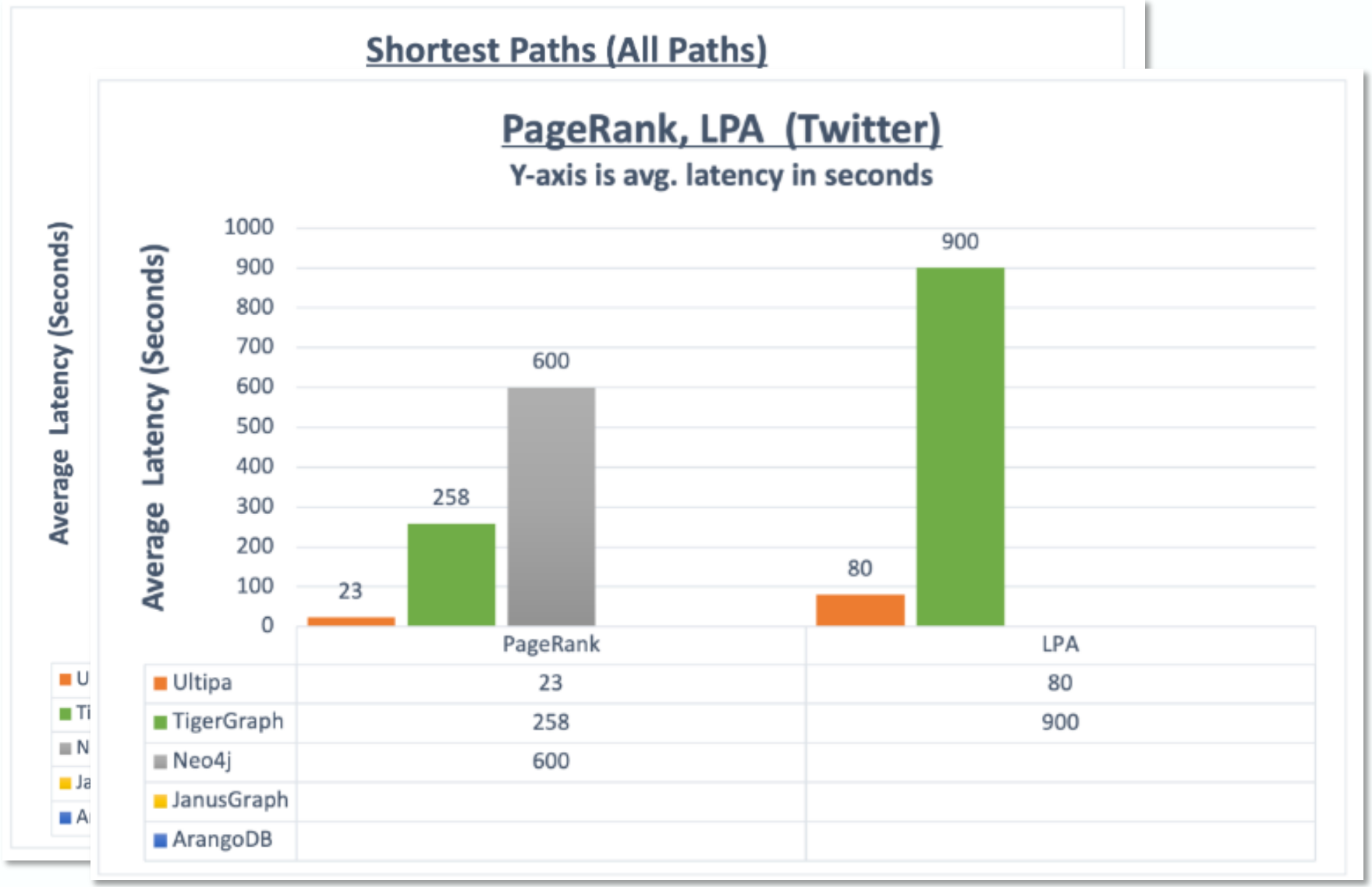
Advanced Algorithms

- Hyper-ANF
- k-NN
- k-Core
- MST
- k-Means
- Clustering Coefficient
- Auto-Networking

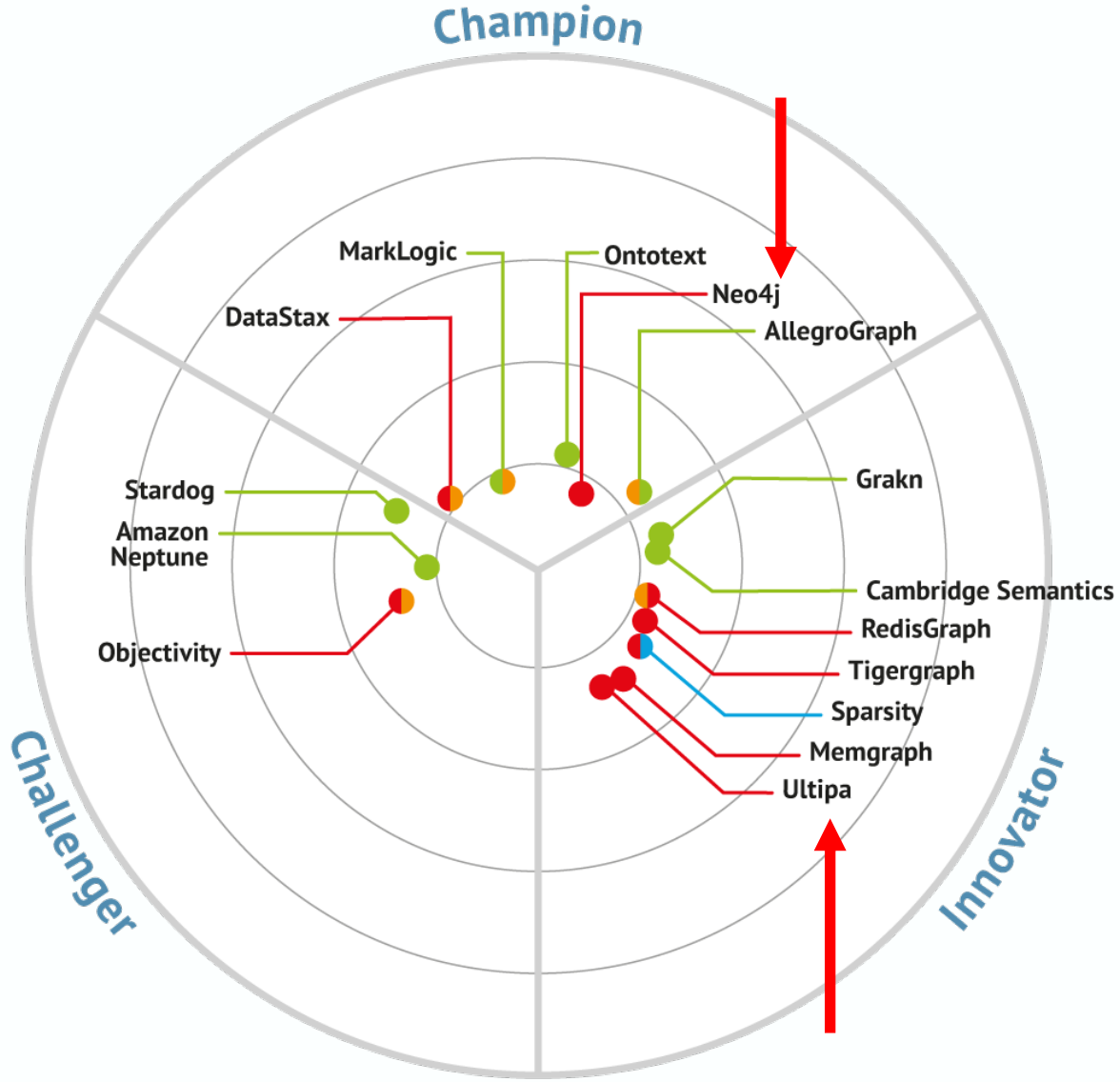
https://www.ultipa.com/document/ultipa-graph-analytics-algorithms/introduction/v4.3

Benchmark

https://www.ultipa.com/article/benchmarks/ultipa-graph-benchmark-report



Not all graph databases are the same!



https://www.bloorresearch.com/technology/graph-databases/

Ultipa & South Korea

- We are looking for partners located in South Korea
- We are open to helping potential customers select the right graph database for their requirements (hopefully Ultipa!). Let's work on a Proof of Concept.
- Are you interested in learning more about Ultipa Graph and becoming an Ultipa Certified Engineer soon? Here's my proposal:
  - I will schedule a free online, exclusive, one-day training for GUG members this November.
  - We are offering a free voucher to obtain certification (a $99 value) for the first 10 people who send a request to yuri.simione@ultipa.com using a business email.

LinkedIn: https://www.linkedin.com/in/yurisimione/
Twitter: https://twitter.com/artika4biz
Email yuri.simione@ultipa.com
Cell. +39 333 4156248

감사합니다