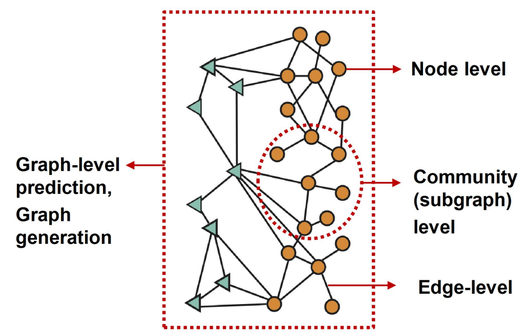
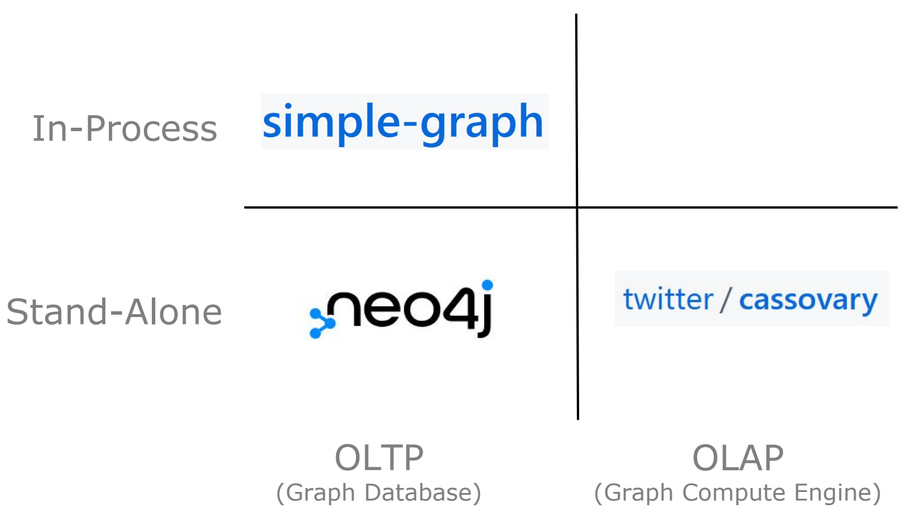
A Graph Database for Data Science Workflows

Data Science & ML Workflows do not only need data management solutions for handling relational data but also solutions for data that are best represented as a graph. Commonly, such workflows on graph data include preprocessing steps for feature engineering on the node, edge, subgraph or graph level and thus require fast variable data access methods.



Source: Leskovec, J., 2021. *Stanford CS224W: Machine Learning with Graphs.*

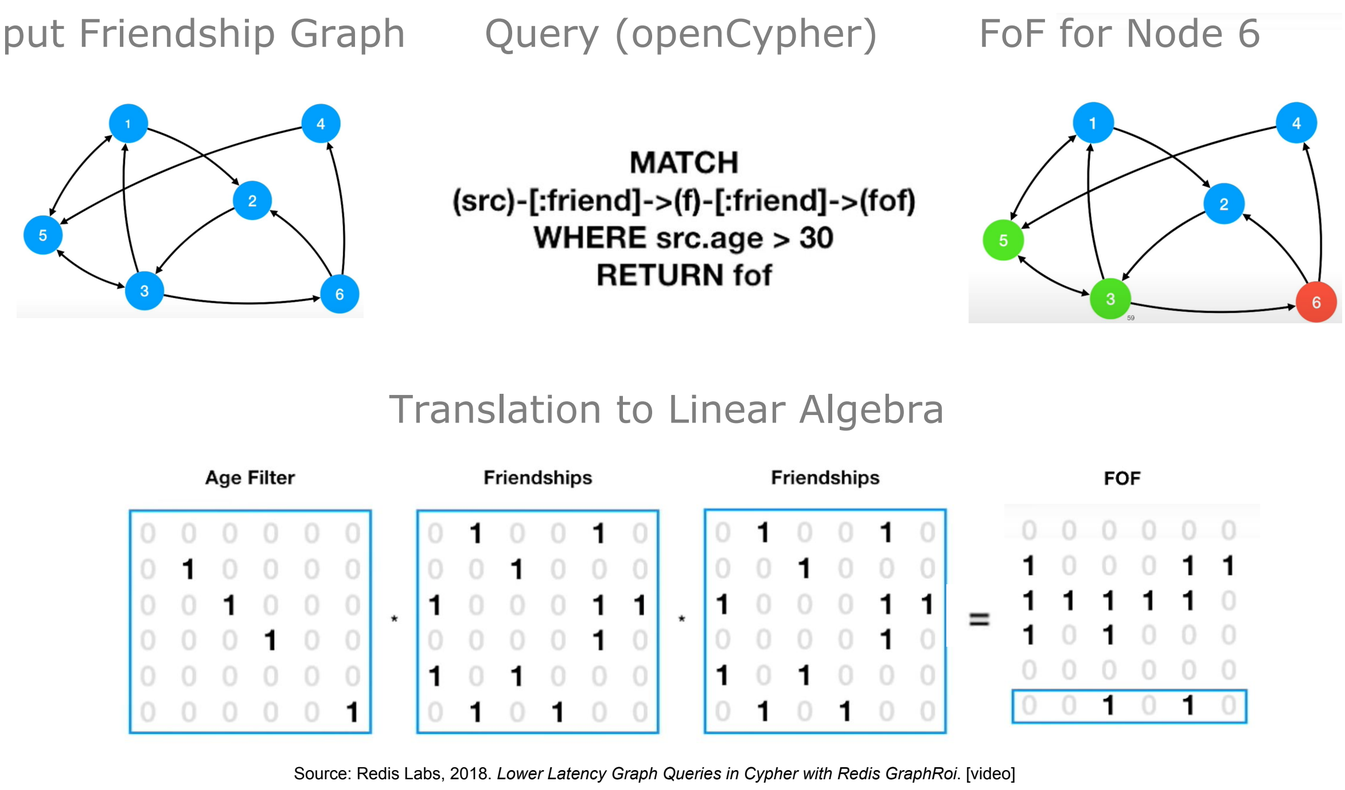


Specialized graph databases have existed for a long time however they do not suit the need of interactive data processing tasks on personal computers with scripting languages such as R or Python. The reason is that most graph databases, similar to RDBMS data warehouse solutions, are designed to run on dedicated server-grade hardware and are built with the client-server architecture which uses inefficient client protocols for data transfer between client and server.

The Goal of this research proposal is to answer the question: **How can we design a database suited for interactive machine learning tasks on graphs?** Since such a database design proposal is a complex undertaking, it needs to be broken down further into subgoals according to a suitable software development framework. The subgoals should include requirement engineering for graph ML pipeline tasks and a review of existing graph database execution engines and their suitability for ML pipeline tasks.

The Solution should not be constrained by untested assumptions, however it is likely that to fulfill the described need for fast local data processing, a shift away from client-server architecture to a small serverless in-process architecture is promising. Two of the main reasons are the advantages of "zero-copy" for speed and "zero configuration" for ease of management. Another consideration is whether to rely on the *GraphBlas* engine to use linear algebra to execute queries on graphs and to store the graphs as adjacency matrices in compressed sparse column format. This is for example used in the *RedisGraph* Database and promises fast execution.

Example: FoF with Matrix Multiplication



Source: Redis Labs, 2018. *Lower Latency Graph Queries in Cypher with Redis GraphRoi.* [video]

Connection to Lecture

This research question is primarily based on the idea behind DuckDB, presented by Dr. Hannes Mühleisen in his lecture on in-process OLAP. DuckDB is a RDBMS motivated by the need for local processing of medium-sized relational data sets in data science workloads. It is described by its author as belonging to a new class of data management systems for embedded analytics. However, as mentioned on their website, DuckDB is not suited for operating on graphs and thus raises the question of how to design the DuckDB equivalent for Graph Databases.

Source: Raasveldt, M, Mühleisen, H.F, 2020. Data Management for Data Science - Towards Embedded Analytics, in: Proceedings of the Conference on Innovative Data Systems Research.

Jonathan Haas

Master Student @ Hasso Plattner Institute, Potsdam, Germany

E-Mail: Jonathan.Haas@student.hpi.uni-potsdam.de

Hasso Plattner Institut