Exploring Large Scholarly Networks with Hermes

Gabriel Campero Durand, Marcus Pinnecke, Anusha Janardhana,
Yusra Shakeel, Jacob Krüger, Thomas Leich, Gunter Saake
Email: *{firstname.lastname@ovgu.de}*

Introduction

The traditional metrics of science (e.g., the H-index), that rely on network-unaware statistics, have been questioned by scholars, making a case for improving the study of scholarly networks by complementing metrics with content & network-based analysis.

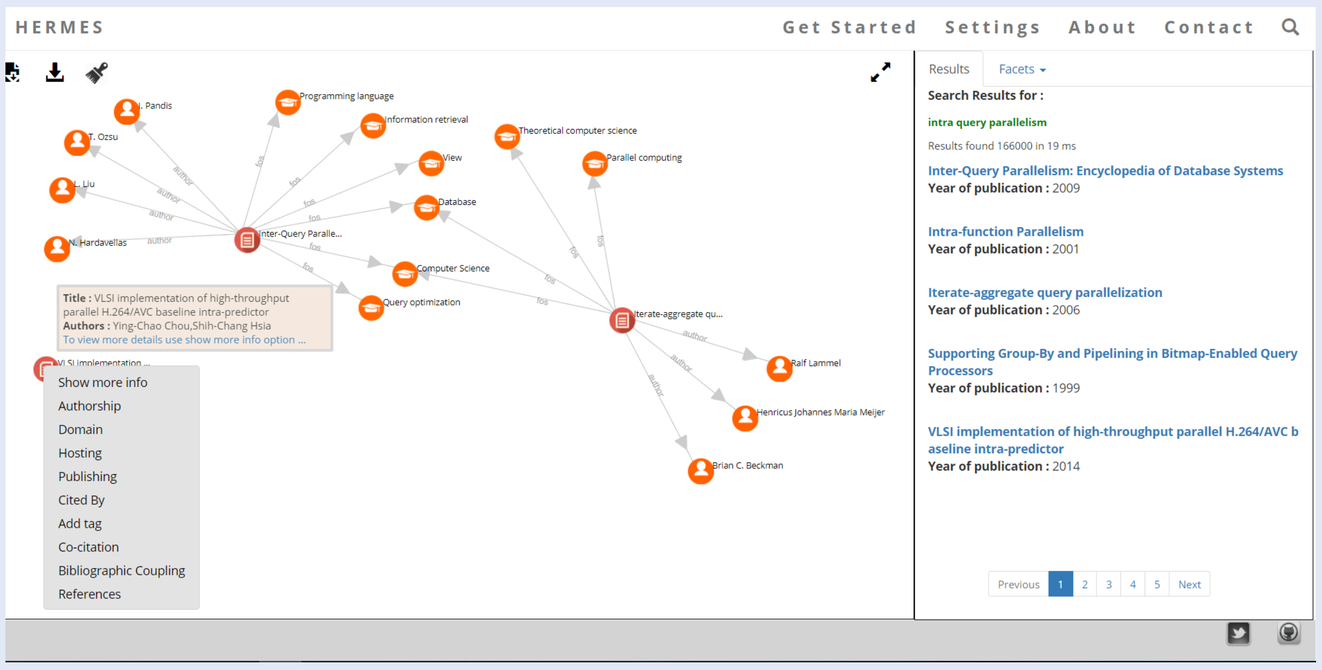

Figure 1: Browsing with Hermes

Hermes is an early stage tool we're developing to support efficient network analysis of large scholarly data in multi-user environments.

In our first approaches, we start from a simple prototype based on JanusGraph, Cassandra and Elasticsearch, and we explore:

- **a closer integration between graph and search engines**
- **incremental processing of temporal queries**
- **extensive use of indexes**

With our tool, we seek to enable users to seamlessly interact with large-scale heterogeneous networks, performing ad-hoc SNA at different granularities, either through our APIs, or through native graph/search-engine query languages.

Workload Characterization

SNA networks at different levels, each with different kinds of queries:

- **macro** (global),
- **meso** (community),
- and **micro** (individual)

Better understanding of workloads can help tuning and provisioning. We intend to use Hermes for such studies.

Networks in Scholarly Data

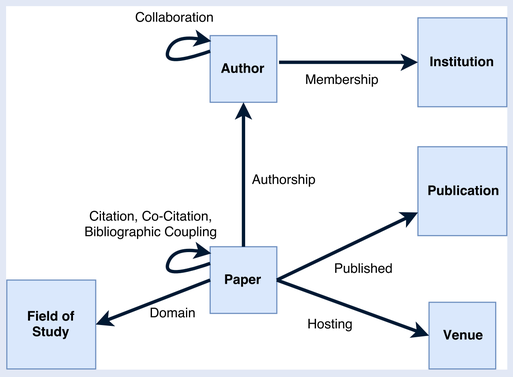

Figure 2: Data Model

At least **7 types of networks are used for SNA**: co-authorship/collaboration, citations, co-citations, bibliographical coupling, topics, co-words, and heterogeneous networks.

Restrictive choices of network type and aggregation entity can limit the generalization power of SNA. To avoid this, **researchers recommend to employ heterogeneous networks, and methods capable of extracting value from them**.

Rewriting Graph to Search Engine Queries

Degree Centrality with Gremlin (Basic Graph):
g.V().in(label).inV().group().by(.ID()).by(.inE(label).count())

```
TermsBuilder termsBuilder = AggregationBuilders.terms(aggregationName).
    field("idOfTargetVertex").size(numberOfVertices);

XContentBuilder contentBuilder;
contentBuilder = JsonXContent.contentBuilder().startObject();
termsBuilder.toXContent(contentBuilder, ToXContent.EMPTY_PARAMS);
contentBuilder.endObject();

SearchRequestBuilder searchRequest= esClient.prepareSearch("index").
    setTypes("edges")).
    setQuery(QueryBuilders.termQuery("edgeLabel","label")).
    setAggregations(contentBuilder);
response = searchRequest.execute().actionGet();
```

Figure 3: Degree Centrality with a Search Engine

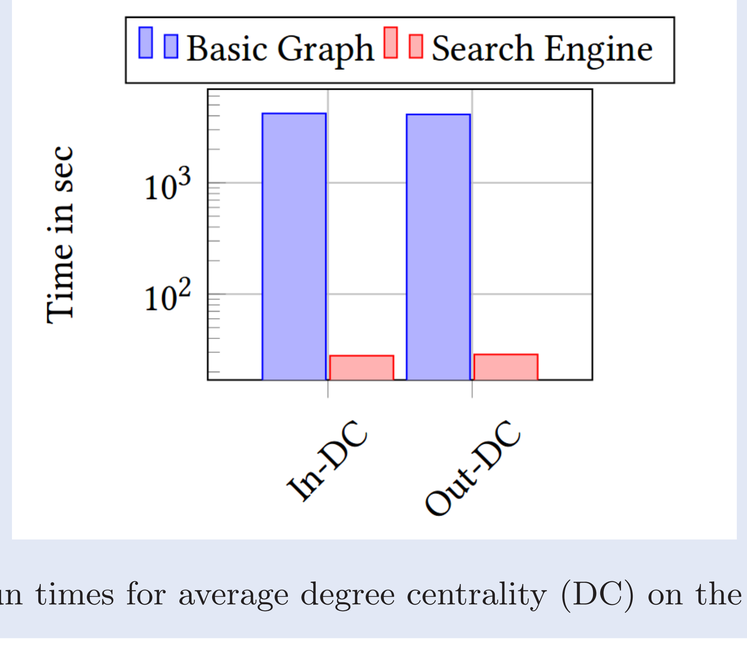

Figure 4: Run times for average degree centrality (DC) on the Pokec dataset

Next Steps and Open Challenges

- We seek to perform user studies with Hermes for workload characterization and improving our tool. We're also taking requests for more functionality.
- We're studying if we can formalize query rewrites across engines, for automating them.
- We aim to migrate Hermes to a specialized storage we're developing within our in-house HTAP DBMS: GeckoDB.
- Open challenges include determining representative usage scenarios, scalable data loading, supporting users in data cleaning, management of analysis results, and integration with external sources.

Acknowledgements

We thank Andreas Meister for revisions. This work is partially funded by DFG grants SA 465/50-1 and LE 3382/2-1, and the DAAD STIBET Matching Funds grant.