Query execution in schema-optional graph databases generally consists of matching regular query patterns against the data graph, an application of the subgraph isomorphism problem. Without building exhaustive indices, such queries are typically far harder to optimize than relational queries, which can be resolved against the database schema and take advantage of its static typing. This is due to the comparatively reduced structure in the storage layouts of schema-optional databases to accommodate polymorphism, which increases the number of unnecessary pointer accesses that need to be performed in order to ensure result correctness, potentially between sparsely located records.

Conventional wisdom advocates for providing as much information as possible in the query pattern to aid optimization; however, we can demonstrate that Cypher behaves inconsistently in practice and, in some cases, removing information from a query actually improves performance. We use the Labelled Subgraph Query Benchmark (LSQB) dataset to show this phenomenon and, in one case, demonstrate that a 3x performance improvement is possible. Particularly, we examine cases where traversals can be pruned from the ends of paths, and where labels can be removed from nodes or relationships, without modifying the semantics of the query. We also show a case where the same techniques instead reduce performance, highlighting the need for selectivity in applying them. Metadata collected from the storage layer has historically been used in query plan selection and has the further potential to guide such cases.

In this workshop, we demonstrate how to modify the Cypher queries without changing query semantics by considering the schema used for generation of the LSQB dataset. We then analyze the resulting query plans produced by Neo4j and compare them to the original query plans to understand how performance is improved or worsened. Finally, we discuss how we repurpose storage layer metadata to drive query optimization by identifying and selectively apply such schema-based optimizations in the absence of a provided schema, a technique that has wide applicability in schema-optional graph databases.