Query Optimizer

GSQL has a built-in query optimizer that can optimize the execution plan of a query when you install the query. As of TigerGraph 3.7, the optimizer can optimize a query in the following ways:

- Cost-based optimization

What is the query optimizer?

The query optimizer optimizes the compiled binary execution plan of a query.

When you install a GSQL query, GSQL compiles the query logic into a binary execution plan. If your query contains a path pattern, the pattern contains a source vertex set, one or more path edge patterns, and one or more step vertex sets. Similar to the path pattern in the GSQL query, its binary execution plan also starts from a set of source vertices, traverses through edges, and finds the target vertices.

However, the actual execution plan of the query might perform path traversals differently than specified in the GSQL query to produce results. When you install a query, the query optimizer can optimize the execution plan of a query so that it executes more efficiently.

Depending on the type of optimization used, this could allow a query to execute with less time, less memory, or less storage. As of 3.7, TigerGraph supports cost-based optimization, which reduces the execution time of a query.

Cost-based optimization

Cost-based optimization reduces time it takes to execute a query.

How it works

The cost-based optimizer takes statistics regarding the vertex and edge types as well as their attributes that are referenced in the query to compile an execution plan that reduces the amount of time it takes to execute a query.

To use the cost-based query optimizer, see Enabling Cost-Based Optimizer.