Dgraph: Graph database for production environment

Tomasz Zdybał, Dgraph Labs

https://dgraph.io

Feb 09, 2017

Overview

1. Introduction
2. Design
3. Highly debated decisions
4. GraphQL+- query language
5. Benchmarks
6. Q & A

Introduction

What is Dgraph?

Dgraph is an open source graph database built for web-scale production environments written entirely in Go.

What is Dgraph?

Dgraph is an open source graph database built for web-scale production environments written entirely in Go.

Goals

- Sharded and Distributed
- Automatic Replication
- Consistent
- Highly Available by design
- Fault Tolerant

Design

```
NQuad:
 Subject  Predicate  Object   .
<alice>   <friend>   <bob>    .
<alice>   <friend>   <eve>    .
<bob>     <friend>   <eve>    .

Posting:
 UID, Value, ValueType, Lang, Label, ...

Posting List:
 Subject Predicate  => Postings...
<alice> <friend>    => [<bob>, <eve>]

Shard:
 All posting lists for one predicate
<friend> => [<alice friend>, <bob friend>]

Group:
 Shards of different predicates form a group
 [*,P1,*], [*,P2,*], ...
```

Raft and Replication

- Groups are replicated across servers.
- Each server can handle many groups.
- Reads and writes go via Raft consensus algorithm.
- Ideally 1, 3 or 5 servers handling a group (odd number for consensus).

Query Execution

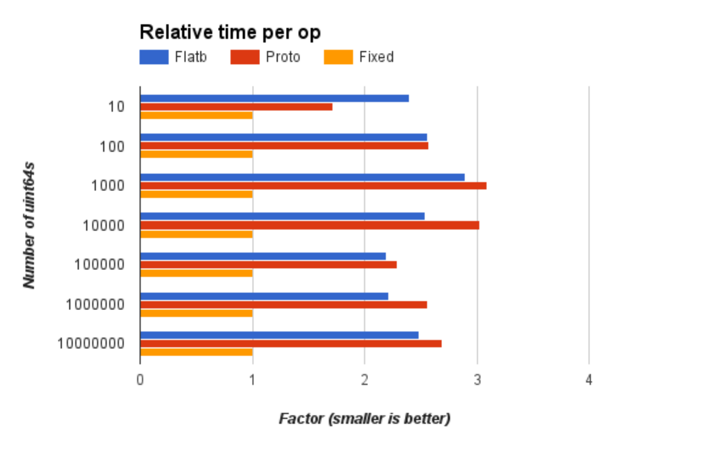
- Tree-like query structure (derived from GraphQL).
- Each branch is handled by one goroutine, allowing concurrent execution of sibling branches.
- Rough algorithm:
    1. Retrieving results from network
    2. Apply Filters, if any [one goroutine per child filter]
    3. If required, do sort over network and run pagination
    4. If only count required, do count and return
    5. Process children, one child per goroutine recursively. Each child again starts at 1.

Highly debated decisions

RocksDB vs BoltDB

- BoltDB acquires a global mutex lock around all reads and writes; which skews it towards read-only system.
- Going to Cgo for RocksDB is as fast as Bolt; but performs a whole lot better under read-write workload.
- Custom Key-Value Store implementation is planned.

Protocol Buffers vs Flatbuffers



GraphQL+- query language

GraphQL+-

- used as primary language of communication
- feature-rich variation of GraphQL
- added support for graph operations
- removed features not fitting well for a graph database

GraphQL+-

- used as primary language of communication
- feature-rich variation of GraphQL
- added support for graph operations
- removed features not fitting well for a graph database

Why GraphQL as a base?

- graph-like query syntax
- schema validation
- subgraph shaped response

GraphQL+- features

- pagination (using *first*, *offset* and *after* arguments)
- aliasing of predicate names
- counting number of entities
- filtering (at root and predicate level)
- various functions (comparison, term matching, geolocation)
- sorting/ordering
- variables
- result normalization

```
{
  me(id: m.06pj8) {
    name.en
    director.film @filter(allof(name.en, "jones indiana") OR
        allof(name.en, "jurassic park")) {
      _uid_
      name.en
    }
  }
}
```

Benchmarks

Cayley

- Cayley is a graph layer, not a graph database.

Cayley

- Cayley is a graph layer, not a graph database.
- Data handling is done outside of Cayley by other databases.

Cayley

- Cayley is a graph layer, not a graph database.
- Data handling is done outside of Cayley by other databases.

Results (Cayley with Bolt on Macbook)

- Loading 21M RDFs – Dgraph is 9.7x faster
- Queries (Gremlin) – Dgraph is 36.6x faster
- Queries (MQL) – Dgraph is 5x faster

Cayley

- Cayley is a graph layer, not a graph database.
- Data handling is done outside of Cayley by other databases.

Results (Cayley with Bolt on Macbook)

- Loading 21M RDFs – Dgraph is 9.7x faster
- Queries (Gremlin) – Dgraph is 36.6x faster
- Queries (MQL) – Dgraph is 5x faster

- Dgraph would perform even better on distributed setup, when distributed joins come into play.
- Link to full benchmark code:
  https://github.com/ankurayadav/graphdb-benchmarks

Neo4j

- Probably the most popular graph database

Neo4j

- Probably the most popular graph database

Results (on Thinkpad T460)

Neo4j

- Probably the most popular graph database

Results (on Thinkpad T460)

- Loading 1.1M RDFs – Dgraph is 100x faster

Neo4j

- Probably the most popular graph database

Results (on Thinkpad T460)

- Loading 1.1M RDFs – Dgraph is 100x faster
- Queries – Dgraph is 3x-6x faster on a read-write workload and at least as fast on read only

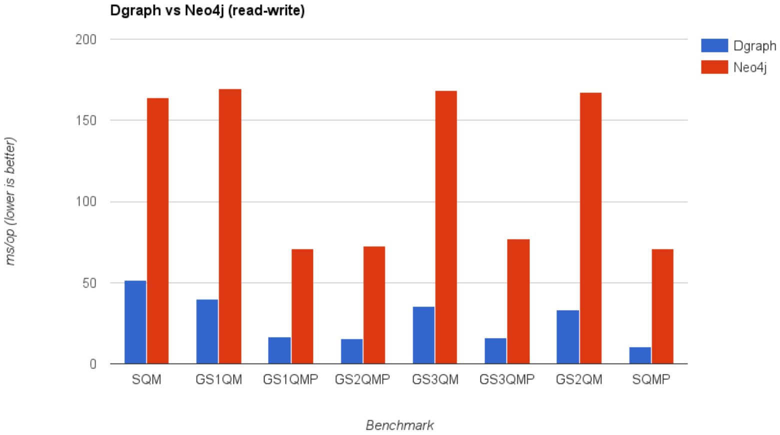
Neo4j

- Probably the most popular graph database

Results (on Thinkpad T460)

- Loading 1.1M RDFs – Dgraph is 100x faster
- Queries – Dgraph is 3x-6x faster on a read-write workload and at least as fast on read only

- Benchmark description: https://open.dgraph.io/post/benchmark-neo4j/
- Link to full benchmark code: https://github.com/dgraph-io/benchmarks/tree/master/data/neo4j


Dgraph vs Neo4j (read-write)
Dgraph
Neo4j
ms/op (lower is better)
200
150
100
50
0
SQM
GS1QM
GS1QMP
GS2QMP
GS3QM
GS3QMP
GS2QM
SQMP
Benchmark

How Dgraph can load up data at least 100 times faster than Neo4j (clickable link)

Q & A

Questions and Answers