High-level query language for Graph databases

Manasi Sunil Bharde (msb4977@rit.edu)
Advisor: Dr. Carlos Rivero
Rochester Institute of Technology

Abstract

- Similar to relational algebra, graph algebra and algorithms can be used to design a system to query specific information, patterns from the graph database.
- Performance of different compositions of query is computed and compared for IMDB dataset loaded to Neo4j

Introduction

- GraphQL is a query language over graph database and uses Graph Algebra [1]. However, GraphQL is incomplete query language. Timber framework is a NoSQL database as it allows heterogeneity in the definition by use of XML files [2].
- Similar to TAX, a query language can be created for graphs which does not flatten the graph tuples and uses graph algebra instead since graphs do not have root node as that for tree.
- Neo4j cannot deal with two different databases having same definition or partial overlapping definition. Hence data is partitioned at high-level by adding a label representing a partition to all nodes belonging to a partition.

Query Operators

- Match: Subgraph matching
- Select .. where: Predicate filtering
- Project : Tuples to return
- Join : Join of disjoint graphs
- Group By : Group nodes by attributes values

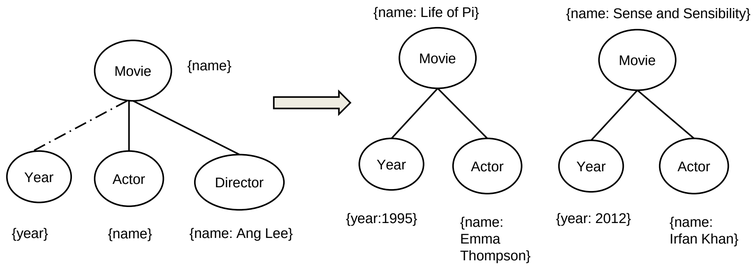

Figure 1. Query graph and matching graph tuples in IMDB database

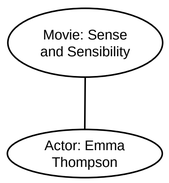

Figure 2. Tuple from Actor database

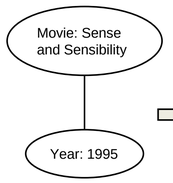

Figure 3. Tuple from Year database

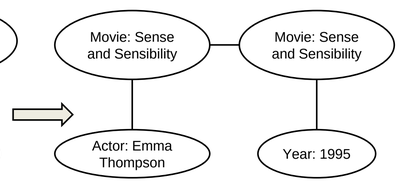

Figure 4. After performing join

Query operator workflows

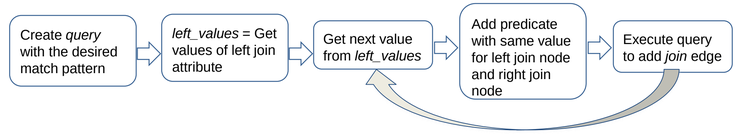

Figure 5: Join operator

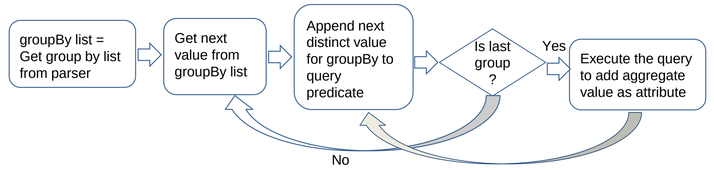

Figure 6: GroupBy operator

Figure 7: Flow with pattern matching prior to predicate

Comparison with cypher query

```
MATCH (a1)--(a2),(a3)--(a4)
WHERE a1:movie AND
 a1:actor-db  AND
a2:actor AND a2:actor-db
AND a3:movie
AND a3:year-db
AND a4:year AND
a4:year-db
AND a1.name='Sense and
Sensibility' AND
a3.name='Sense and Sensibility'
AND a2.name='Emma Thompson'
RETURN a1,a2, a3,a4
```

⇒

```
SELECT m1 movie a actor
FROM actor-db
SELECT m2 movie y year
FROM year-db
PROJECT a m1 y
MATCH
m1 a
m2 y
WHERE
a.name= "Emma Thompson"
JOIN
actor-db.m1.name = year-
db.m2.name
```

References

[1] Huahai He and Ambuj K. Singh. 2008. Graphs-at-a-time: query language and access methods for graph databases. In Proceedings of the 2008 ACM SIGMOD international conference on Management of data (SIGMOD '08). ACM, New York, NY, USA, 405-418. DOI: https://doi.org/10.1145/1376616.1376660

[2] H. V. Jagadish, L. V. Lakshmanan, D. Srivastava, and K. Thompson, “Tax:A tree algebra for xml,” inDBPL, vol. 1. Springer, 2001, pp. 149–164.

Results

Table 1 : Results with optimizer based on number of nodes covered by predicate. Predicates are ordered in decreasing order of span of predicate. Filtering with predicates and then matching pattern is found costlier than matching pattern and then filtering.

Number of Nodes	6	5	2
Number of Relations	8	4	2
Predicate then match	12879 ms	13351 ms	7167 ms
Match then predicate	3665 ms	3925 ms	3826 ms
Complete cypher query	4012 ms	3057 ms	2551 ms

Table 2: Results with optimizer based on selectivity of the predicate. Predicates are ordered in decreasing order of selectivity. Filtering with predicates and then matching pattern is found costlier than matching pattern and then filtering.

Number of Nodes	6	5	2
Number of Relations	8	4	2
Predicate then match	36699 ms	35037 ms	11032 ms
Match then predicate	3926 ms	4085 ms	4107 ms
Complete cypher query	3693 ms	3360 ms	2485 ms

Discussion

- Join operation: As seen in Figure 5, Join operation allows addition of new edges representing join. Join on condition eliminates the need of performing cartesian product of disconnected tuples. It is cartesian product that degrades performance of Neo4j.
- GroupBy operation: In proposed query language, group by operator enables a user to specify the label and attribute value to calculate aggregation of some other attribute value.

Conclusions

- Relabeling a result in the database allows reordering of operations in the query without using extra memory or computation. Hence relabeling is used to compare the performance of different order of query operations.
- Performing subgraph matching before applying predicate always gives better performance over applying predicate and then performing subgraph matching. This is the result of the linked list like storage of data and weak indexing support for attributes in Neo4j.
- Query optimization based on the selectivity of a predicate is slower than query optimization based on span of predicate since the earlier involves cost of calculating distinct values for attribute in the predicate.