Query Optimization in the RG Framework

CHONG FENG U4943054
SUPERVISED BY DR. QING WANG
Outline

1. Background
2. RG Architecture
3. Research Problem
4. Methodology
5. Environment
6. Evaluation
RG (Relation-Graph) Framework

- Provides an unified platform for **network analytics**, developed by Minjian Liu under Dr. Qing’s supervision
- Uses a **hybrid data model** (relations and graphs)
- Defines a SQL-like query language by extending SQL with graph constructing operations
- Incorporates the **RG engine**, which is built upon PostgreSQL using Python
Architecture of the RG Engine

Diagram from “Towards a Unified Framework for Network Analytics”, by Minjian Liu
Research Problem

How to improve the performance of RG engine?

• Network analysis is often *computationally expensive*
• Relational query optimizer exists in the architecture but *graph query optimizer* does not
• Develop a mechanism that can *reuse* and *rewrite* queries to improve its efficiency
• Develop a mechanism that can *estimate* cost and choose the best execution plan
Methodology-Step1: Query Reusing

- A RG-SQL query is made of **relational** queries and **graph** operations.
- The optimizer will use a hashing algorithm to generate a **hash value** for RG queries and sub-queries.
- RG query execution results can be **cached** in the memory or disk to reduce computation for future query processing.
Methodology-Step2: Algorithm Recommendation

- Graph operations are implemented in **multiple** algorithms
- Different algorithms have different **usages** and can generate different **outputs**
- RG engine optimizer will be able to help user to select the most **suitable algorithm** based on historical data and graph characteristics
- RG engine will provide APIs to help incorporate more algorithms
Methodology-Step3: Query Rewriting

- The optimizer will determine whether it is possible and advantageous to **rewrite** the original RG-SQL query and **restructure** the plan tree.
- The optimizer will enumerate alternative plans and estimate cost based on **statistics** from the database metadata.
- It will choose the plan tree with the **lowest cost** to proceed.
Evaluation & Environment

Datasets

- Twitter Dataset:
  - 17,069,982 users
  - 476,553,560 tweets

- Stack Overflow Dataset:
  - 19,824,320 users
  - 7,214,697 questions

Measurements

- Time
- Memory usage
- Disk usage

Environment

- Ubuntu 14.04.4 LTS
- Python 2.7.9
- PostgreSQL 9.5
- Psycopg 2.6.1
Thank you!

CHONG FENG U4943054@ANU.EDU.AU
0425060886