Integrated Cache Based NLP Analysis

By Jiaxi Lei and Dennis Wu

1. Introduction

Typical news media covers an exceptionally large amount of text, so conducting NLP analysis requires the database and query language to have high efficiency. The traditional relational database with optimization may handle storage properly, but some essential operations in NLP analysis such as table joining and repetitive row iteration may still be unsatisfactory. Hence, our group thought of Neo4j, a graph database, as a means to deliver such resource-expensive operations. Also, the query operations on Neo4j can be further optimized by Redis, a key-value pair based database, to implement the temporal based cache. With the help of a graph database like Neo4j, we are able to populate the result as multisets. The advantage of storing the result of joining cities and legislators tabular table into a graph database is the network, which enables a more direct relationship and more efficient recursive query.

2. Project Details

- initialize the project structure:
    - the repo with the basic data scientific structure has been created
- the languages to be used: python
- databases to be used:
    - PostgreSQL as the base for storing the tabular data of legislator.csv, cities.csv, and usnewspapersample.csv
    - Redis as the query database
    - Neo4j as the target database for result collection and query presenting

- steps:
    1. construct the docker that initialize the needed environment variables
    2. populate the PostgreSQL database with the tabular datasets in traditional PK structure

Table	PK	Columns
legislator	(first name, last name)	
cities	(state, city_name) filter out all the US cities, drop dupe	
usnewspaper	(id)	

    3. implement cache on with Redis on PostgreSQL
        a. Key: state(s), Value: all data from PostgreSQL related to state(s)
        b. function inputs are cached
        c. TTL based cache
    4. NLP analysis of newspaper
        a. Extract corpus as metadata for news articles
        b. Populate Neo4j with news article metadata

    5. Populate Neo4j with selected data from (3), implement Redis based cache to query Neo4j graph db and setup vertices by the number of occurrences. The vertices come with labels corresponding to the PostgreSQL table-name, and edges to be the number of occurrences that the two nodes are mentioned in the same document.
    6. The query function should accept keywords from cities and/or legislators, with the following situations:
        a. Given a single input, analyze the keyword's neighborhood with a user-specified degree, default to 3.
        a. Given two inputs, find their all shortest paths.
        b. Given multiple inputs, show a strongly connected component?

    populate Postgres ⇒ NLP analysis on newspaper ⇒ Populate Neo4j ⇒ Queried on Redis

3. Result

The result should be a graph or a list of vertices.

4. References

Redis as cache: Using Redis as an LRU cache – Redis

Strongly connected component:

https://neo4j.com/docs/graph-algorithms/current/labs-algorithms/strongly-connected-components/