Michigan State University
CSE 480 Honor Project Proposal
Graph Database Simulator

Hanghang Zhang A47035780

February 3, 2017

Abstract

Graph database like Neo4j is a very powerful tool which helps researchers building a better understanding of the inner relationship among different data. Unlike a traditional database, a graph database stores the information of data in form of graph structures. Because its special structure, many complicate machine learning models can be implemented within database with relatively simple query language. And building a graph database simulator could be very helpful for my future studying in database and data science.

0.1 Project Summary

For this project, I plan to build a neo4j simulator with python3. The simulator should be able to accept input queries in form of Cypher, and then execute those queries in a virtual database. Graphic result will be returned in form of png. But the result will be a graph that only represents the relationship of each node, users can not interact with it like using a real neo4j database. And I suppose I can finish the whole project within the rest of this semester individually.

0.2 Technical Approaches

In the initial plan, some open source packages will be used to finish this project. Probably networkx and matplotlib will be used to present result graph upon users' search queries. And sqlite3 will be used to simulate a regular sql database. The version control part will be done with github.
The main idea of this project is building the whole thing base on a traditional database, and three models will be implemented to do this:
1. A model that translate graph structure into regular data frame.
2. A model that parse Cypher query into corresponded SQL query.
3. A model that represent data in database in form of png image.
If this project can be finished few weeks before the end of semester, I will try to build a front end gui for this project with PyQt5.

0.3 Project Plan

Planned deadline for each model:
By Feb 22, finished the translation model.
By Mar 20, finished the parser model.
By Apr 24, finished the represent model.