Protein Data Querying and Analysis

-Sowmya Janmahanthi, Vani Pant, Devika Mini Chenkilathu

Project Overview

This project focuses on querying and analyzing protein data using MongoDB (a document-based NoSQL database) and Neo4j (a graph database). The primary objectives are to store, construct, and annotate protein data for advanced querying, visualization, and analysis.

Dataset Information

- **Organism**: *Arabidopsis thaliana*
- **Rows and Columns**: 16390x8
- **Souce:** UniProt

MongoDB: Document Store for Protein Data

Raw protein data is stored in MongoDB as a TSV file. The data includes InterPro domains, which are split into different strings. Jaccard similarity is calculated and all imported along with three separate files into MongoDB for advanced querying.

Key Attributes

- Entry (ID), Entry Name, Protein Name, Gene Names, EC Number, InterPro

Key MongoDB Queries: Counting Total Proteins, Retrieving Labeled and Unlabeled Proteins(based on existence of EC number)

Big Graph ConstructionA Protein-Protein Network (PPN) is constructed based on domain composition.

- **Nodes**: Proteins with taxonomy, sequence, and EC numbers.
- **Edges**: Weighted by domain similarity (Jaccard similarity).

  Key Neo4j Queries: Creating Nodes, Linking the created indexes, and edge representation for the following output.

Output

- A weighted, undirected graph in Neo4j.
- Labeled Nodes: Proteins containing EC numbers.
- Unlabeled Nodes: Proteins awaiting annotation.

Annotation Implementation

Goal: Annotate unlabelled proteins using highest similarity of the neighbour proteins

The Cypher query for annotation is used in Neo4j. Later, In GUI we can search for a particular protein (e.g., A0A0A7EPL0), unlabelled proteins in its neighbourhood (e.g., F4I9T0) are annotated with EC numbers.

GUI Implementation

Using Python and required packages, a web interface is created to interact with the protein dataset.

Features:

1. **Protein Details**: Search and retrieve detailed information about a specific protein.
2. **Graph Visualization**: Display the graph construction of a particular protein, including annotations and attributes.
3. **Neighbor Details**: Show information about neighboring proteins.
4. **Statistics**: Data fetched dynamically from MongoDB and Neo4j.
    - Count of labeled and unlabeled proteins (before and after annotation).

Conclusion

This project demonstrates a robust framework for querying, visualizing, and analyzing protein data. Key contributions include:

- Integration of MongoDB for document-based storage and querying.
- Neo4j-based graph construction and annotation using Jaccard similarity.
- A user-friendly GUI for seamless interaction and insights into protein datasets.