Implementing an Algorithm for matching Biological Graphs: VF2 and VF2 Plus

Shraddha Atrailwalkar (ssa2923@rit.edu)
Advisor: Carlos Rivero
Rochester Institute of Technology

INTRODUCTION

- Graphs are best suited data structures for representing entities and their relationships. Often, there arises a need to find pattern of one graph in another graph (especially in biological graphs).
- Need to implement efficient algorithms that detect whether one graph is isomorphic to another.

BACKGROUND

- Previous implementations like graphQL only detected sub graph isomorphism but did not determine the matching sub graphs.
- They also implemented a recursive DFS approach which turned out to be computationally expensive for larger graphs.

GOALS

- To implement an efficient graph matching algorithm.
  - Implement the subgraph matching algorithm VF2 and an improvement over it, VF2 Plus.
  - Compare VF2 and VF2 Plus using different sizes of biological graphs.

IMPLEMENTATION DESIGN

- The dataset contains Molecules, Contact Maps and Proteins. The dataset contains target and query files with details of graphs (nodes,labels,edges) and ground truth files containing matches between target and query graphs.
- The graphs were visualized using JGRAPHX libraries and JGRAPHWT was used to generate graph objects from above mentioned graph files.
- The implementation consisted of the following modules:
  - Parser: Takes an input a graph file, parses it and creates graph objects.
  - Algorithm Implementation : VF2
    - Depends on structural information provided by the graph
    - Implements feasibility rules to find matching nodes in the graph
    - Feasibility rules (n & m from query; n' & m' from target):
      - $R_{pred}$/ $R_{succ}$ (if n is a predecessor/successor of m in g1, then n' is a predecessor/successor of m'),
      - $R_{term}$, $R_{termout}$ (Number of pred/succ of n in (inbound/outbound) terminal set for both graphs is equal)
    - $R_{new}$ (No. of pred/succ of n & n' neither in mapping/terminal set is equal)
  - Algorithm Implementation: VF2 Plus
    - Computes the probability for each node using the structural information
    - Sorts the search space from the most to least probable node in solution set
    - Generates smaller subset of nodes, applies feasibility rules on each subset, discards the ones that do not follow, thereby reducing the search space.

Result: Comparison with ground truth files: Results for a pair of graphs are compared to those already present in the ground truth files. The performance of each algorithm on the same set of graphs is also analyzed to understand which implementation is better.

SCENARIO

Consider the query(g1) and target(g2) graph. VF2 and VF2 Plus are implemented to find matches between the 2 graphs:

VF2 starts with one node from each graph and recursively checks every pair for feasibility:
- $T$: g2.grf
- $P$: g1.sub.grf
- N: 1
- S: 1,3,2,4,8,4,1,0,5,7

RESULTS

- The figure shows the graph after parsing the .grf files of query and pattern graphs.
- The files are parsed and JGRAPHWT library is used to create graph objects which are used by each algorithm for computation.
- JGRAPHX library is used to visualize the graph.

REFERENCES

[1] Artem Maksov, Yong Li, Reese Butler, 'Subgraph Isomorphism' published March 04, 2015, The University of Tennessee
[2] Vincenzo Carletti, Pasquale Foggia, and Mario Vento, ’VF2 Plus: An Improved version of VF2 for Biological Graphs', University of Salerno, Salerno, Italy