Improving Subgraph Isomorphism by Postponing Cartesian Product

Ishan D Gulhane (idg5922@rit.edu) | Advisor: Dr. Carlos Rivero
Rochester Institute of Technology

Introduction
- Graphs are widely used in social networks and proximity algorithms.
- Two graphs G and H are said to be isomorphic if there is a bijection which preserves the adjacency and non-adjacency.

The goal of this project is to implement and analyze algorithms for improving speed of subgraph isomorphism.

The size of the search space can be reduced by leveraging the following information:
- Using local pruning and global structural information to remove infeasible mappings of query node.
- Leveraging vertex information to generate adapted subgraph for query graph to improve the speed of matching.
- Order the matching query nodes to reduce the size of Cartesian product, thus reducing the size of search space.

To perform subgraph matching, we process the candidate set for each query node containing all nodes in graph which satisfies the properties of given query node and then prune the false positive nodes using different techniques.

To search the embedding of query graph, nodes are traversed in particular sequence which reduces the number of join operation performed.

Algorithms
- Brute Force Implementation:
  - Find all the feasible candidates for each node in subgraph.
  - The resulting product of all the mappings forms the search space.
- Refine Search Space using Bipartite matching:
  - Pseudo subgraph isomorphism using Bipartite matching.
  - Search Order Generation: A search order is represented as a binary tree with internal nodes as a joint operation and leaf nodes are the nodes of the pattern graph.

Subgraph Isomorphism by postponing Cartesian product:
- Develop a compact auxiliary path based data structure, i.e., Compact Path Index (CPI)
  - Top Down Construction: Visit the query nodes in level order and prune the false positive node mappings by using the non-tree edges and edges with previous level performed in two steps: Forward Candidate Generation and Backward Candidate Pruning.
  - Bottom Up Refinement: Prunes the mappings based on the presence of edges with the child nodes, i.e., nodes present in the next level of BFS of query graph.

Results
- Brute Force Implementation:
  - Find all the feasible candidates for each node in subgraph.
  - The resulting product of all the mappings forms the search space.
- Refine Search Space using Bipartite matching:
  - Pseudo subgraph isomorphism using Bipartite matching.
  - Search Order Generation: A search order is represented as a binary tree with internal nodes as a joint operation and leaf nodes are the nodes of the pattern graph.

Subgraph Isomorphism by postponing Cartesian product:
- Develop a compact auxiliary path based data structure, i.e., Compact Path Index (CPI)
  - Top Down Construction: Visit the query nodes in level order and prune the false positive node mappings by using the non-tree edges and edges with previous level performed in two steps: Forward Candidate Generation and Backward Candidate Pruning.
  - Bottom Up Refinement: Prunes the mappings based on the presence of edges with the child nodes, i.e., nodes present in the next level of BFS of query graph.

Future Work
- The future work involves developing new algorithms which leverage the use of already computed CPI of queries that are part of other queries.

References
- Huahai He, Ami B, Singh. "Graphs-at-a-time: Query Language and Access Methods for Graph Databases".
- Fei Bi, Lijun Chang, Xueming Lin, Lu Qin, Wenjie Zhang "Efficient Subgraph Matching by Postponing Cartesian Products".