**Motivation**

With the increasing popularity of Web large network data can be retrieved to study the evolution of social networks and information flow at microscopic level. The evolution of such network data can be represented in one single structure (time evolving graph) since the data in these networks does not disappear with time but rather grows.

**Subgraph Isomorphism**

- Support calculation requires solving subgraph isomorphism problem.
- Current implementation of GERM uses Ullmann’s algorithm (Ullmann, 1976).

**Definitions**

- Frequent based pattern is a pattern with support larger than certain threshold
- Support is a non-negative anti-monotonic function of P
- Minimum image based support
- Graph evolution rule body = head
- Body = iterative time pattern without edges with maximum timestamps
- Support of graph evolution rule = support of head
- Minimum image based support (P, T)
- Color of the nodes = labels
- DFS Code encodes a pattern, but not uniquely

**Citation Network**

- Citation network has papers as nodes and citations as edges. If a paper cites paper j, there is a directed edge from node representing paper j to node representing paper i.
- Two major datasets are considered:
  - SWIR dataset which contains papers from major conferences in Semantic Web and Information Retrieval and was crawled from CiteSeerX and DBLP (Balak, Kamiskki and Hayes, 2011)
  - Nodes: 10971
  - Edges: 25481
  - Years: 2001 - 2007
  - Node labels: keywords extracted from full text
- Papers in the graph are assigned timestamps to edges and assigns labels to nodes. Time evolving graph structure needs to be managed efficiently.

**GSpan**

- Gspan is developed for transactional setting (Yen and Han, 2002)
- By constructing DFS Code Tree patterns are taken from the original graph and not generated randomly
- By introducing DFS Lexicographic Order search space is efficiently pruned
- Given graph G=(V,E,L):
  - V is the set of nodes
  - E is the set of edges
  - L is a label function over nodes and edges
- Problem Statement:
  - Given graph G=(V,E,L):
  - Time evolving graph G
  - Assigns timestamps to edges
  - Assigns labels to nodes
  - The task is to find frequent based evolution rules in the graph G

**DFS Code**

- DFS Code is a \( \lambda (I_i, N_i) \) where \( \lambda (I_i, N_i) \) is a lexicographic combination of DFS order over edges and full order over node labels
- DFS Lexicographic Order is a lexicographic combination of DFS order over edges and full order over node labels
- DFS Code Tree encodes a pattern,
- But not uniquely

**GERM vs neoGERM**

- GERM adapts GSpan to one single graph setting
- GERM works only for undirected graphs
- GERM requires data to be stored as document
- GERM allows only integer numbers as labels over nodes
- neoGERM requires data to be stored in Neo4j and incorporates in-built methods from Neo4j
- neoGERM works for directed graphs
- neoGERM works with node labels as set of integers and node labels for partial matching
- neoGERM further improves support calculation

**Related Work**

- Microscopic evolution: (Leskovec, Backstrom, Kumar, Tomkins, 2008)
- Link prediction algorithms like Adamic-Adar/Liben-Nowell and Kleinberg are capable to predict links between existing nodes
- Experiment on different datasets with different settings.
- Further improve support calculation.
- Develop link prediction based upon discovered graph evolution rules.
- Investigate neoGERM scalability and performance on larger datasets.

**Future Work**

**Initial Results**

- Experimental results on different datasets with different settings.
- Further improve support calculation.
- Develop link prediction based upon discovered graph evolution rules.
- Investigate neoGERM scalability and performance on larger datasets.

**Citation Network**

- Citation network has papers as nodes and citations as edges. If a paper cites paper j, there is a directed edge from node representing paper j to node representing paper i.
- Two major datasets are considered:
  - SWIR dataset which contains papers from major conferences in Semantic Web and Information Retrieval and was crawled from CiteSeerX and DBLP (Balak, Kamiskki and Hayes, 2011)
  - Nodes: 10971
  - Edges: 25481
  - Years: 2001 - 2007
  - Node labels: keywords extracted from full text
- Papers in the graph are assigned timestamps to edges and assigns labels to nodes. Time evolving graph structure needs to be managed efficiently.

**Neo4j**

Neo4j is a NoSQL graph database (www.neo4j.org).
- It has a traversal framework for high speed traversals on the nodes.
- Gephi has a plug-in to visualise databases from Neo4j.
- Neo4j is an open source project in a GPLv3 Community edition with Advanced and Enterprise editions available under both AGPLv3 and commercial licenses.

**Subgraph Isomorphism**

- Support calculation requires solving subgraph isomorphism problem.
- Current implementation of GERM uses Ullmann’s algorithm (Ullmann, 1976).

**Definitions**

- Frequent based pattern is a pattern with support larger than certain threshold
- Support is a non-negative anti-monotonic function of P
- Minimum image based support
- Graph evolution rule body = head
- Body = iterative time pattern without edges with maximum timestamps
- Support of graph evolution rule = support of head
- Minimum image based support (P, T)
- Color of the nodes = labels
- DFS Code encodes a pattern, but not uniquely

**Citation Network**

- Citation network has papers as nodes and citations as edges. If a paper cites paper j, there is a directed edge from node representing paper j to node representing paper i.
- Two major datasets are considered:
  - SWIR dataset which contains papers from major conferences in Semantic Web and Information Retrieval and was crawled from CiteSeerX and DBLP (Balak, Kamiskki and Hayes, 2011)
  - Nodes: 10971
  - Edges: 25481
  - Years: 2001 - 2007
  - Node labels: keywords extracted from full text
- Papers in the graph are assigned timestamps to edges and assigns labels to nodes. Time evolving graph structure needs to be managed efficiently.

**GSpan**

- Gspan is developed for transactional setting (Yen and Han, 2002)
- By constructing DFS Code Tree patterns are taken from the original graph and not generated randomly
- By introducing DFS Lexicographic Order search space is efficiently pruned
- Given graph G=(V,E,L):
  - V is the set of nodes
  - E is the set of edges
  - L is a label function over nodes and edges
- Problem Statement:
  - Given graph G=(V,E,L):
  - Time evolving graph G
  - Assigns timestamps to edges
  - Assigns labels to nodes
  - The task is to find frequent based evolution rules in the graph G

**DFS Code**

- DFS Code is a \( \lambda (I_i, N_i) \) where \( \lambda (I_i, N_i) \) is a lexicographic combination of DFS order over edges and full order over node labels
- DFS Lexicographic Order is a lexicographic combination of DFS order over edges and full order over node labels
- DFS Code Tree encodes a pattern,
- But not uniquely

**GERM vs neoGERM**

- GERM adapts GSpan to one single graph setting
- GERM works only for undirected graphs
- GERM requires data to be stored as document
- GERM allows only integer numbers as labels over nodes
- neoGERM requires data to be stored in Neo4j and incorporates in-built methods from Neo4j
- neoGERM works for directed graphs
- neoGERM works with node labels as set of integers and node labels for partial matching
- neoGERM further improves support calculation

**Related Work**

- Microscopic evolution: (Leskovec, Backstrom, Kumar, Tomkins, 2008)
- Link prediction algorithms like Adamic-Adar/Liben-Nowell and Kleinberg are capable to predict links between existing nodes
- Experiment on different datasets with different settings.
- Further improve support calculation.
- Develop link prediction based upon discovered graph evolution rules.
- Investigate neoGERM scalability and performance on larger datasets.

**Future Work**

**Initial Results**

- Experimental results on different datasets with different settings.
- Further improve support calculation.
- Develop link prediction based upon discovered graph evolution rules.
- Investigate neoGERM scalability and performance on larger datasets.