Mining Generalized Graph Patterns based on User Examples

Pavel Dmitriev, Carl Lagoze

Department of Computer Science
Cornell University, Ithaca, NY, USA
Outline

- Background and Motivation
- Related Work
- Our Approach
- Experimental Results
- Conclusion and Future Work
Outline

- Background and Motivation
- Related Work
- Our Approach
- Experimental Results
- Conclusion and Future Work
A lot of recent interest in extracting useful information from graphs

Typically, information = subgraphs/patterns
- Clustering protein interaction and gene co-expression networks
- Discovering web communities
- Classifying chemical compounds

Often, the exact structure of subgraphs of interest is not known
Motivation

- Popular approach: use frequent subgraphs
- However, often interested in finding subgraphs that perform specific function in the network
  - Studying evolution of functional modules in biological networks
  - Identifying logical documents on the Web
- The subgraphs/patterns of interest are not necessarily frequent
Problem Definition

- Assume subgraphs of interest are generated from a set of **Cores**, or basic subgraphs, according to some **Generalization Rules (GRs)**
- Call such subgraphs **Generalized Patterns (GPs)**

- Problem (most general):
  
  Given a graph, find all GPs in the graph
Example

Generalized Patterns

Generalized Patterns

Core

Generalization Rule

P1

P2

P3

P4
Outline

- Background and Motivation
- Related Work
- Our Approach
- Experimental Results
- Conclusion and Future Work
Related Work

- *Hofer et al, 2004*. Mining molecular fragments with wildcards

- *Meinl et al, 2004*. Mining frequent molecular structures containing chains of atoms of varying length

- *Kashtan et al, 2004*. Role-based generalization rules applicable to arbitrary subgraphs
Outline

- Background and Motivation
- Related Work
- Our Approach
- Experimental Results
- Conclusion and Future Work
Our Approach

- Machine learning approach:
  - User specifies GPs of interest in the graph
  - Cores are generated based on user examples and their context in the graph
  - GRs are created based on the structure of the cores
  - Find embeddings of cores in the graph, and use GRs to obtain GPs from these embeddings
Our Approach

- Machine learning approach:
  - User specifies GPs of interest in the graph
  - Cores are generated based on user examples and their context in the graph
  - GRs are created based on the structure of the cores
Example
Positive and Negative Examples
Core Generation
Core Generation
Our Approach

- **Machine learning approach:**
  - User specifies GPs of interest in the graph
  - Cores are generated based on user examples and their context in the graph
  - GRs are created based on the structure of the cores
Natural Expansion Rule

Natural Expansion Rule: Given a core \( c \) of size \( m \), and a subgraph \( g \), NER produces a graph \( g' \) by adding a copy of \( c \) to \( g \) in such a way that \( m-1 \) vertices are taken from \( g \), and only one new vertex is introduced.

Proposition: Given an arbitrary graph \( g \), a core \( c \), and an embedding \( s \) of \( c \) in \( g \), there exists a unique GP \( p \) induced from \( c \) in \( g \) by the NER, containing \( s \).
Example
Theorem

- Neg$_u =$ negative examples generated from user-provided GPs
- Neg = negative examples generated from all GPs
- k = max. allowed size of a negative example
- **Theorem:** If GPs in the graph do not overlap, and (1) Neg$_u$ = Neg; (2) no user example matches an example from Neg$_u$; (3) max size of a core is $\leq k$, then every GP produced by the algorithm is a subgraph of a real GP.
Outline

- Background and Motivation
- Related Work
- Our Approach
- Experimental Results
- Conclusion and Future Work
Experimental Setup

**WP₁**
- 3 – 10 vertices

**WP₂**
- 2 – 10 vertices

**BP₁**
- 3–8 vertices, role is chosen at random, strong/weak GR

**BP₂**
- 4–8 vertices, role is chosen at random, strong/weak GR
Results

- When the number of added links is small (below 50), performance is good for all patterns.

- As the number of added links increases:
  - Performance for WP₂ is still good.
  - Performance for WP₁ decreases.
  - Performance for biological patterns is reasonably good, but the variance is high.
Conclusion and Future Work

- Formulate a GP mining problem as a machine learning problem, which puts much less burden on the user, and present an algorithm to solve it.
- Verified the effectiveness of the algorithm on synthetic data generated using real world patterns from biological and web domains.

**Future work:** use user help (active learning) to improve core generation, address performance issues, apply to real-world problems.
Thank you!

Long version of the paper can be downloaded from:

http://www.cs.cornell.edu/~dmitriev/