Correlation Search in Graph Databases

Yiping Ke James Cheng Wilfred Ng

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology

Outline

1. Introduction
2. Problem Definition
3. Solution
4. Performance Evaluation
5. Conclusions

Introduction

Graphs

- Model objects and their relationships
- Everywhere in various scientific domains
  - Bioinformactics: protein interaction networks
  - Chemistry: chemical compound structures
  - Social science: social networks
  - Many more: work flows, Web site structures, etc

Existing Research on Graph Search

- Focus on structural similarity search: find the graphs structurally the same as or similar to a given query graph

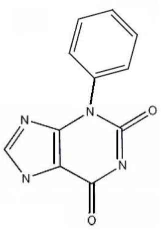
Introduction

Graphs

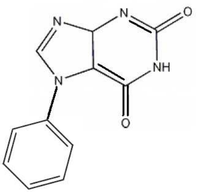
- Model objects and their relationships
- Everywhere in various scientific domains
  - Bioinformactics: protein interaction networks
  - Chemistry: chemical compound structures
  - Social science: social networks
  - Many more: work flows, Web site structures, etc

Existing Research on Graph Search

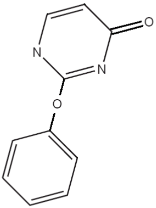
- Focus on structural similarity search: find the graphs structurally the same as or similar to a given query graph



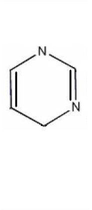
(a) Graph A



(b) Graph B



(c) Graph C

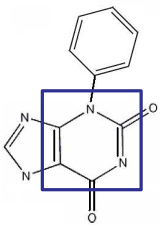


(d) Query

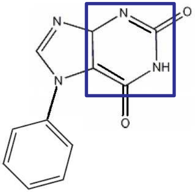
Application Scenario

- Structural similarity search
- Need: find co-occurrent molecular structure of a given molecule
- Structural similarity search fails to find such results
- Co-occurrent structures may decide some chemical properties

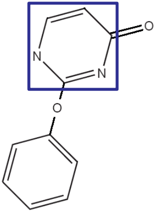
Introduction - Motivation



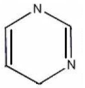
(a) Graph A



(b) Graph B



(c) Graph C

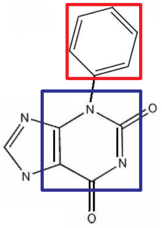


(d) Query

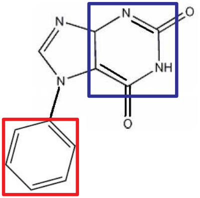
Application Scenario

- Structural similarity search
- Need: find co-occurrent molecular structure of a given molecule
- Structural similarity search fails to find such results
- Co-occurrent structures may decide some chemical properties

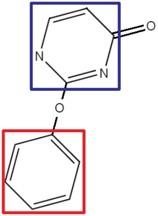
Introduction - Motivation



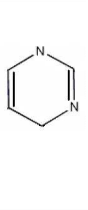
(a) Graph A



(b) Graph B



(c) Graph C

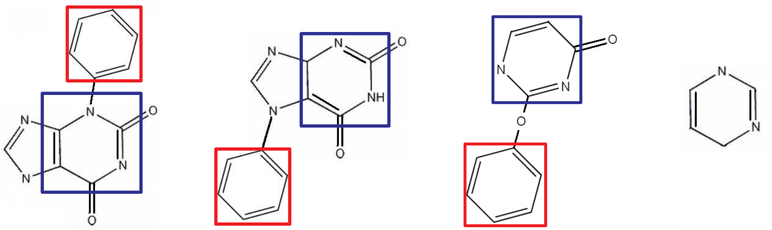


(d) Query

Application Scenario

- Structural similarity search
- Need: find co-occurrent molecular structure of a given molecule
- Structural similarity search fails to find such results
- Co-occurrent structures may decide some chemical properties

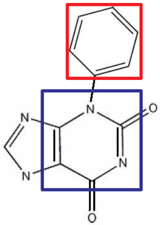
Introduction - Motivation



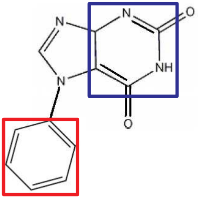
(a) Graph A (b) Graph B (c) Graph C (d) Query

Application Scenario

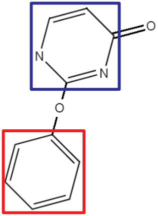
- Structural similarity search
- Need: find co-occurrent molecular structure of a given molecule
- Structural similarity search fails to find such results
- Co-occurrent structures may decide some chemical properties



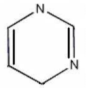
(a) Graph A



(b) Graph B



(c) Graph C



(d) Query

Application Scenario

- Structural similarity search
- Need: find co-occurrent molecular structure of a given molecule
- Structural similarity search fails to find such results
- Co-occurrent structures may decide some chemical properties

Introduction - Correlation Search in Graph Databases

Correlation

- Capture the underlying dependence between objects
- Well-studied in boolean databases, quantitative databases, multimedia databases, data streams, and many more

New Challenges in Graph Databases

- Large search space
  - Each subgraph of a graph in the database is a candidate
  - Exponentially many subgraphs
- Expensive graph operation
  - Subgraph isomorphism testing (NP-Complete)

Introduction - Correlation Search in Graph Databases

Correlation

- Capture the underlying dependence between objects
- Well-studied in boolean databases, quantitative databases, multimedia databases, data streams, and many more

New Challenges in Graph Databases

- Large search space
  - Each subgraph of a graph in the database is a candidate
  - Exponentially many subgraphs
- Expensive graph operation
  - Subgraph isomorphism testing (NP-Complete)

Introduction - Correlation Search in Graph Databases

Correlation

- Capture the underlying dependence between objects
- Well-studied in boolean databases, quantitative databases, multimedia databases, data streams, and many more

New Challenges in Graph Databases

- Large search space
  - Each subgraph of a graph in the database is a candidate
  - Exponentially many subgraphs
- Expensive graph operation
  - Subgraph isomorphism testing (NP-Complete)

Introduction - Correlation Search in Graph Databases

Correlation

- Capture the underlying dependence between objects
- Well-studied in boolean databases, quantitative databases, multimedia databases, data streams, and many more

New Challenges in Graph Databases

- Large search space
  - Each subgraph of a graph in the database is a candidate
  - Exponentially many subgraphs
- Expensive graph operation
  - Subgraph isomorphism testing (NP-Complete)

Introduction - Contributions

New Problem of Correlated Graph Search (CGS)

- Correlation measure: Pearson's correlation coefficient

Effective and Efficient Solution: CGSearch

- Theoretical bounds for the support (occurrence probability) of a candidate
- Candidate generation from the projected database of query graph
- Three heuristic rules to further reduce number of candidates

Introduction - Contributions

New Problem of Correlated Graph Search (CGS)

- Correlation measure: Pearson's correlation coefficient

Effective and Efficient Solution: CGSearch

- Theoretical bounds for the support (occurrence probability) of a candidate
- Candidate generation from the projected database of query graph
- Three heuristic rules to further reduce number of candidates

Problem Definition - Correlation Measure

Pearson's Correlation Coefficient

Popularly used as a correlation measure in many other contexts: stream data, transaction databases

Definition

$$\phi(g_1, g_2) = \frac{supp(g_1, g_2) - supp(g_1)supp(g_2)}{\sqrt{supp(g_1)supp(g_2)(1 - supp(g_1))(1 - supp(g_2))}}$$

- Measure the departure of two variables from independence
- Fall within $[-1, 1]$: 0 indicates independence; positive indicates positive correlation; negative indicates negative correlation
- Our work: focus on positive correlation

Problem Definition - Correlation Measure

Pearson's Correlation Coefficient

Popularly used as a correlation measure in many other contexts: stream data, transaction databases

Definition

$$\phi(g_1, g_2) = \frac{supp(g_1, g_2) - supp(g_1)supp(g_2)}{\sqrt{supp(g_1)supp(g_2)(1 - supp(g_1))(1 - supp(g_2))}}$$

- Measure the departure of two variables from independence
- Fall within $[-1, 1]$: 0 indicates independence; positive indicates positive correlation; negative indicates negative correlation
- Our work: focus on positive correlation

Problem Definition - Correlation Measure

Pearson's Correlation Coefficient

Popularly used as a correlation measure in many other contexts: stream data, transaction databases

Definition

$$\phi(g_1, g_2) = \frac{supp(g_1, g_2) - supp(g_1)supp(g_2)}{\sqrt{supp(g_1)supp(g_2)(1 - supp(g_1))(1 - supp(g_2))}}$$

- Measure the departure of two variables from independence
- Fall within $[-1, 1]$: 0 indicates independence; positive indicates positive correlation; negative indicates negative correlation
- Our work: focus on positive correlation

Problem Definition

CGS Problem

Given a graph database $\mathcal{D}$, a correlation query graph q and a minimum correlation threshold θ $(0 < \theta \leq 1)$, find the set of all graphs whose Pearson's correlation coefficient with q is no less than θ

Solution - Candidate Generation

Bounds of *supp*(*g*)

$$\frac{supp(q)}{\theta^{-2}(1 - supp(q)) + supp(q)} \leq supp(g) \leq \frac{supp(q)}{\theta^{2}(1 - supp(q)) + supp(q)}$$

Range: Candidate Generation from $\mathcal{D}$

Mine the set of Frequent subGraphs (FGs) from $\mathcal{D}$ using the above two bounds as thresholds

Drawback

- All existing FG mining algorithms generate graphs with higher support before those with lower support
- Not efficient and scalable, especially when $\mathcal{D}$ is large or the lower bound is low

Solution - Candidate Generation

Bounds of *supp*(*g*)

$$\frac{supp(q)}{\theta^{-2}(1 - supp(q)) + supp(q)} \leq supp(g) \leq \frac{supp(q)}{\theta^{2}(1 - supp(q)) + supp(q)}$$

Range: Candidate Generation from $\mathcal{D}$

Mine the set of Frequent subGraphs (FGs) from $\mathcal{D}$ using the above two bounds as thresholds

Drawback

- All existing FG mining algorithms generate graphs with higher support before those with lower support
- Not efficient and scalable, especially when $\mathcal{D}$ is large or the lower bound is low

Solution - Candidate Generation

Bounds of $supp(g)$

$$\frac{supp(q)}{\theta^{-2}(1 - supp(q)) + supp(q)} \leq supp(g) \leq \frac{supp(q)}{\theta^{2}(1 - supp(q)) + supp(q)}$$

Range: Candidate Generation from $\mathcal{D}$

Mine the set of Frequent subGraphs (FGs) from $\mathcal{D}$ using the above two bounds as thresholds

Drawback

- All existing FG mining algorithms generate graphs with higher support before those with lower support
- Not efficient and scalable, especially when $\mathcal{D}$ is large or the lower bound is low

Solution - Candidate Generation (cont')

Bound of $supp(q, g; \mathcal{D}_q)$

$$supp(q, g; \mathcal{D}_q) \geq \frac{1}{\theta^{-2}(1 - supp(q)) + supp(q)}$$

Candidate Generation from $\mathcal{D}_q$

Mine the set of FGs from $\mathcal{D}_q$ using the above threshold

Compared with Range

- $\mathcal{D}_q$ is much smaller than $\mathcal{D}$
- The minimum support threshold is higher

Advantages

- Efficient candidate generation
- Significant reduction in search space

Solution - Candidate Generation (cont')

Bound of $supp(q, g; \mathcal{D}_q)$

$$supp(q, g; \mathcal{D}_q) \geq \frac{1}{\theta^{-2}(1 - supp(q)) + supp(q)}$$

Candidate Generation from $\mathcal{D}_q$

Mine the set of FGs from $\mathcal{D}_q$ using the above threshold

Compared with Range

- $\mathcal{D}_q$ is much smaller than $\mathcal{D}$
- The minimum support threshold is higher

Advantages

- Efficient candidate generation
- Significant reduction in search space

Solution - Candidate Generation (cont')

Bound of $supp(q, g; \mathcal{D}_q)$

$$supp(q, g; \mathcal{D}_q) \geq \frac{1}{\theta^{-2}(1 - supp(q)) + supp(q)}$$

Candidate Generation from $\mathcal{D}_q$

Mine the set of FGs from $\mathcal{D}_q$ using the above threshold

Compared with Range

- $\mathcal{D}_q$ is much smaller than $\mathcal{D}$
- The minimum support threshold is higher

Advantages

- Efficient candidate generation
- Significant reduction in search space

Solution - Candidate Generation (cont')

Bound of $supp(q, g; \mathcal{D}_q)$

$$supp(q, g; \mathcal{D}_q) \geq \frac{1}{\theta^{-2}(1 - supp(q)) + supp(q)}$$

Candidate Generation from $\mathcal{D}_q$

Mine the set of FGs from $\mathcal{D}_q$ using the above threshold

Compared with Range

- $\mathcal{D}_q$ is much smaller than $\mathcal{D}$
- The minimum support threshold is higher

Advantages

- Efficient candidate generation
- Significant reduction in search space

Solution - Heuristic Rules

Heuristic 1: identify graphs that are guaranteed to be answers

All supergraphs of q in the candidate set are in the answer set

Heuristics 2 and 3: get rid of false-positives

If a graph g is not in the answer set, prune all its subgraphs that have the same support as q or have support less than $(\theta\sqrt{\frac{(1-supp(q))supp(g)(1-supp(g))}{supp(q)}} + supp(g))$ in $\mathcal{D}_q$

Solution - Heuristic Rules

Heuristic 1: identify graphs that are guaranteed to be answers

All supergraphs of q in the candidate set are in the answer set

Heuristics 2 and 3: get rid of false-positives

If a graph g is not in the answer set, prune all its subgraphs that have the same support as q or have support less than $(\theta\sqrt{\frac{(1-supp(q))supp(g)(1-supp(g))}{supp(q)}} + supp(g))$ in $\mathcal{D}_q$

Solution - CGSearch Algorithm

Input: Graph database $\mathcal{D}$, query q, correlation threshold θ
Output: The answer set $\mathcal{A}_q$

1. Obtain $\mathcal{D}_q$
2. Mine the set of candidate graphs $\mathcal{C}$ from $\mathcal{D}_q$, using $\frac{1}{\theta^{-2}(1-supp(q))+supp(q)}$ as the minimum support threshold
3. Check whether $\phi(q, g) \geq \theta$ for each graph $g \in \mathcal{C}$; refine $\mathcal{C}$ by three heuristic rules

Performance Evaluation

Datasets

- Real dataset: 100K compound structures of cancer and AIDS data, averagely 21 nodes and 23 edges in each graph, 88 distinct labels
- Synthetic dataset: four datasets of 100K graphs by varying average number of edges from 40 to 100, 30 distinct labels and 0.15 average graph density

Other Algorithms Used

- Obtain projected database: FG-index [SIGMOD'07]
- Mine FGs: gSpan [Yan and Han, ICDM'02]

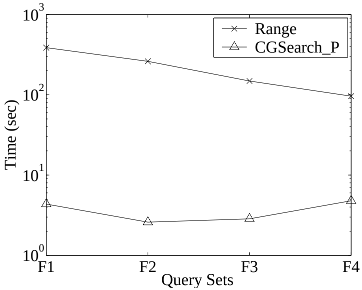
Baseline

- Range: candidate generation from $\mathcal{D}$ with a support range

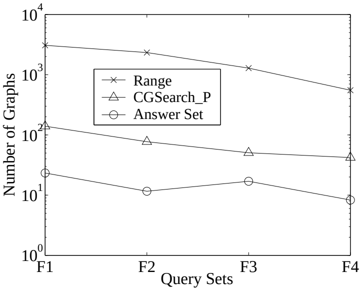
Effect of Candidate Generation when Varying Query Support

Summary

- CGSearch is two orders of magnitude faster than Range
- The candidate set produced by CGSearch is much closer to the answer set and is over an order of magnitude smaller than Range



(a) Running Time

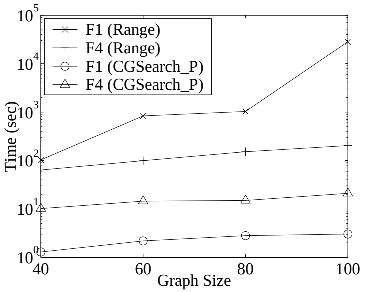


(b) Size of Candidate Set

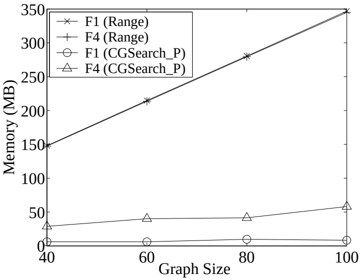
Effect of Graph Size

Summary

- CGSearch is up to four orders of magnitude faster and consumes 41 times less memory than Range
- CGSearch is much more stable on resource usage than Range



(a) Running Time



(b) Memory Consumption

Conclusions

Correlated Graph Search

Take into account the occurrence distributions of graphs using Pearson's correlation coefficient

Mining Algorithm: CGSearch

- Theoretical bounds for support of candidates
- Candidate generation from a projected database
- Three heuristic rules

Experiments

- Candidate generation from the projected database is efficient
- Three heuristic rules are effective
- Compared with *Range*, CGSearch is orders of magnitude faster
- CGSearch achieves very stable performance for various query support, minimum correlation thresholds, as well as graph sizes

Thank you

Q & A

Poster: Board 2 on Aug 13