Assignment 1: q3

Discriminative Subgraph Indexing for Efficient Subgraph Query Processing

1. Introduction

This report presents a novel graph indexing approach designed to accelerate subgraph query processing in large graph databases. Unlike supervised methods that rely on class labels, our technique identifies structurally discriminative subgraphs based solely on their pruning effectiveness, making it applicable to diverse unsupervised graph mining scenarios.

2. Overview

The core objective is to develop a graph indexing scheme that significantly reduces candidate set sizes during subgraph query processing while strictly maintaining false-negative safety—ensuring no true matches are incorrectly filtered out. Since class labels are unavailable in our setting, we reinterpret "discriminative subgraphs" in terms of filtering capability rather than classification accuracy. Our method selects patterns that optimally balance three key properties: coverage (appearing in sufficient graphs), selectivity (not appearing in too many graphs), and structural richness (encoding meaningful graph topology).

3. Methodology

3.1 Subgraph Enumeration Strategy

To generate candidate subgraphs, we perform depth-first search (DFS) based enumeration on each database graph, focusing exclusively on connected subgraphs with up to four nodes. This size restriction manages combinatorial explosion while still capturing meaningful structural patterns. Disconnected subgraphs are excluded as they provide limited structural constraint value.

Each enumerated subgraph undergoes canonicalization to ensure isomorphism invariance:

1. Nodes are sorted by their label and degree
2. Nodes are renumbered deterministically
3. Edges are sorted lexicographically
4. This process ensures that structurally identical subgraphs from different database graphs map to identical canonical string representations.

3.2 Pattern Frequency Analysis

For each canonical pattern p, we compute its support:

$$f(p) = \frac{\text{number of database graphs containing } p}{|D|}$$

Patterns with extreme frequencies ($f(p) < 0.05$ or $f(p) > 0.95$) are discarded as they offer minimal pruning utility—very rare patterns are too specific to be useful, while very common patterns fail to differentiate between graphs.

3.3 Discriminative Scoring Framework

Without class labels for traditional discriminative metrics, we define a pattern's discriminativeness through its filtering potential. Each pattern is evaluated using three complementary components:

Filtering Power: Patterns with frequencies near 0.5 maximize pruning effectiveness:

$$\text{FilteringPower}(p) = 1 - 2|f(p) - 0.5|$$

Structural Complexity: Larger patterns offer greater selectivity but risk becoming overly specific. We cap this contribution to maintain balance:

$$\text{ComplexityFactor}(p) = \min\left(\frac{|V_p| + |E_p|}{5}, 1\right)$$

Information Entropy: This captures the uncertainty reduction provided by knowing whether a pattern appears in a graph:

$$H(p) = -f(p)\log f(p) - (1 - f(p))\log(1 - f(p))$$

The final composite score combines these factors multiplicatively:

Patterns are ranked by this score, and the top-k patterns are selected for indexing.

3.4 Adaptive Pattern Selection

Prior to final pattern selection, we perform an adaptive analysis of the pattern frequency distribution across the database. This identifies how many patterns fall within the moderate frequency range [0.1, 0.9], providing diagnostic guidance for setting k. The system suggests an appropriate k value based on this distribution, but ultimately leaves this parameter user - controllable for domain-specific tuning. This adaptive approach ensures robustness across datasets with varying structural characteristics.

3.5 Index Construction and Query Processing

Each selected pattern defines a binary feature in the index. Database graphs are represented as feature vectors indicating the presence or absence of each indexed pattern. This representation guarantees that for any matching graph, its feature vector contains (as a superset) the features present in the query—a property essential for maintaining false-negative safety.

During candidate generation, we employ soft filtering to balance pruning effectiveness with robustness:

- For a query with m active features, a database graph is retained if it matches at least $\max(1, \lfloor m \cdot \rho \rfloor)$ features
- Here, ρ is a tunable overlap ratio parameter

This relaxed matching criterion prevents eliminating true matches due to missing optional subgraphs while still achieving substantial candidate set reduction.

This relaxed matching criterion prevents eliminating true matches due to missing optional subgraphs while still achieving substantial candidate set reduction.

4. Why This Approach Works: Intuitive Explanation

The effectiveness of our indexing strategy stems from its alignment with the fundamental goal of candidate pruning rather than classification. In unsupervised settings, a subgraph proves most valuable when its presence meaningfully narrows the search space without excluding legitimate matches.

Patterns with moderate frequencies strike an optimal balance—they appear frequently enough to be relevant for many queries, yet remain selective enough to eliminate substantial portions of non-matching graphs. This contrasts with extreme-frequency patterns that either offer no discriminative value (if too common) or insufficient coverage (if too rare).

Incorporating structural complexity ensures selected patterns represent meaningful graph topology rather than trivial node configurations. While more complex subgraphs provide stronger structural constraints, bounding their influence prevents overspecialization to rare, database-specific configurations.

The soft overlap constraint during candidate generation provides crucial robustness. By requiring only partial feature overlap rather than complete containment, the system accommodates structural variations and optional substructures in real-world graphs, avoiding false negatives while maintaining strong filtering performance.

5. Conclusion

Our discriminative subgraph indexing approach offers a principled, unsupervised method for accelerating subgraph query processing. By focusing on patterns that optimize filtering effectiveness through balanced frequency, structural richness, and information content, the method achieves significant candidate set reduction while preserving false-negative safety. The adaptive selection mechanism and tunable parameters provide flexibility across diverse graph databases, making this approach both practical and theoretically sound for large-scale graph search applications.