Arya: Arbitrary Graph Pattern Mining with Decomposition-based Sampling

Zeying Zhu*, Kan Wu*, Alan Zaoxing Liu
Graph-structured Data are Ubiquitous

Social networks
Twitter graph: ten billion edges*

Protein-protein graph
~80 billion nodes and 250 million edges

*GraphJet: Real-Time Content Recommendations at Twitter
Pattern Mining is An Important Analytics Task

• Social networks
  • Spot communities and advertise to users

• Biology
  • Characterize protein-protein structures or interactions

• Finance
  • Money laundering detection

A simple pattern example

Small deposits followed by a large withdrawal
Graph Pattern Mining

Find all subgraph instances matching a given pattern of interest.

Counting the number of any subgraphs
Iterate every isomorphic subgraph
Exact Mining Solutions

Iterate every isomorphic subgraph

Exponentially growing intermediate candidate sets

Optimizations: reduce redundant enumeration, system optimizations, hardware accelerators, but still **NP-Complete**
Scalability Challenge in Exact Mining

Arabesque (SOSP ‘15)
Motifs with size = 3
~1 billion
11 hours

GraphPi (SC ‘21)
ΔΔ Pattern with size = 6
~1.2 billion
>24 hours
Approximate Pattern Mining

• Many mining tasks do not need exact answers.
  • Output density of certain patterns
• List some but not all subgraphs for large graphs.
  • Output representative ones

General approximate approach:
Sample a subset of the input data and estimate the count based on the probability.
Using Neighborhood Sampling [ASAP, OSDI’18]

Sample a subset of the input data and estimate count based on probability

\[ p_1 = \frac{1}{8} \times \frac{1}{4} = \frac{1}{32} \]
\[ e_1 = 32 \]

\[ p_2 = \frac{1}{8} \times \frac{1}{4} = \frac{1}{32} \]
\[ e_2 = 0 \]

\[ p_n = \frac{1}{8} \times \frac{1}{4} = \frac{1}{32} \]
\[ e_n = 0 \]

\[ \frac{1}{n} \sum_{i=1}^{n} e_i \approx 4 \]

Graph

Pattern

Neighborhood sampling

Probability

\[ \Delta \text{ Counting} \]

Result

• Up to 258x speed up in ASAP
ASAP Cannot Scale to Complex Patterns

- Need larger number of samplers for more complex patterns.
- From 4-node to 5-node patterns, there is a $O(\Delta)$ increase.
- $\Delta$ is the maximum degree of the graph.
If the patterns are more complex, it needs significantly more samplers and is less scalable.

Can we reduce the complexity of the sampled pattern?

Our key idea is to leverage graph decomposition theory and sample different sub-patterns individually.
**A powerful theorem** (informal) [S. Assadi et al., 2019]: Solving an optimal fractional edge cover can decompose any patterns into a unique collection of odd cycles and stars, which meets optimal bounds for sampling arbitrary patterns.
Sample Individual Sub-patterns

- Odd cycle sampler: edge sampling

- Star sampler
Form a Pattern

Test remaining edges in 5Cycle-5Star.
Accelerate the computation

Test remaining edges in 5Cycle-5Star.

System optimizations:
- Failure-probability-aware sampler scheduling
- Cache and reuse sampled sub-patterns

(check more details in our paper)
The “Failure Probability” of Cycle/Star Samplers

• Different subpatterns have different sampling failure probabilities.
  • “Failure”: A sampler does not find the pattern

- More likely to fail (e.g., 92%)
- Do not fail

• The subpattern sampling order matters for mining time!
  • If any subpattern sampling fails, the entire pattern sampling fails.
  • We can early terminate the sampling if there are any failures.
Scheduling More-likely-to-fail Subpatterns First

0% fail

100% sample triangle

92% fail

8% sample 1-star

Improve the performance by 2x without affecting accuracy.
Putting It Together -- Arya

1. Graph size (1B)
   Pattern (5house, 5%, 0.95)

2. Fractional Edge Cover
   LP Solver
   Pattern Decomposer

3. Error-Latency Profile
   / User Configuration

4. Count: 12345 +/- 617
   Time: 12s

5. Sampled subgraphs
Evaluation

• Distributed system implementation (11K LOC)
  • OpenMP/MPI
  • Memcached key-value store

• Evaluated on medium, large, and giant Graphs
  • Mico, 1M edges
  • YouTube, 2.9M edges
  • Twitter, 1.2B edges
  • Friendster, 1.8B edges
  • RMAT-5B, 5B edges
  • RMAT-10B, 10B edges

• Patterns
  • 3-Motifs (2 patterns), 4-Motifs (6 patterns), complex patterns (>= 5 nodes)
Evaluation: Exact Mining Systems

Distributed replicated graph setting

<table>
<thead>
<tr>
<th>Platform</th>
<th>Fractal</th>
<th>GraphPi</th>
<th>Arya</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mico</td>
<td>1822</td>
<td>6.3</td>
<td>0.8</td>
</tr>
<tr>
<td>YouTube</td>
<td>2479</td>
<td>36</td>
<td>18</td>
</tr>
<tr>
<td>Twitter</td>
<td>failed</td>
<td>265</td>
<td></td>
</tr>
</tbody>
</table>

- Failed >24h
- 330x

Time (sec)
### Evaluation: Exact Mining Systems

#### Distributed replicated graph setting

<table>
<thead>
<tr>
<th>Time (sec)</th>
<th>Fractal</th>
<th>GraphPi</th>
<th>Arya</th>
</tr>
</thead>
<tbody>
<tr>
<td>1822</td>
<td>6.3</td>
<td>0.8</td>
<td>1</td>
</tr>
<tr>
<td>2479</td>
<td>140x</td>
<td>36</td>
<td>18</td>
</tr>
<tr>
<td>failed &gt;24h</td>
<td>330x</td>
<td>265</td>
<td></td>
</tr>
</tbody>
</table>

- Up to 20,000x faster than Fractal
- Up to 1,000x faster than GraphPi
Evaluation: Approximate Mining Systems

• Arya’s number of samplers is smaller than or similar as ASAP.
• Each Arya’s sampler runs faster because of edge sampling.
Discussion and Future Work

- Sampling-based approaches are hard to find a pattern when the graph is sparse.
- Extending Arya to trillion-edge graph scenarios.
- Selecting the best graph pattern mining algorithm for different graph-pattern inputs.
Conclusions

• Graph pattern mining is important and challenging.
  • Larger and denser graphs and complex and arbitrary patterns.
  • Poor scalability of existing systems.

• Arya leverages graph decomposition theory and sampling techniques for fast and scalable pattern mining.
  • Outperforming existing exact and approximate pattern mining solutions by up to five orders of magnitude.

• Open-sourced at https://github.com/Froot-NetSys/Arya.