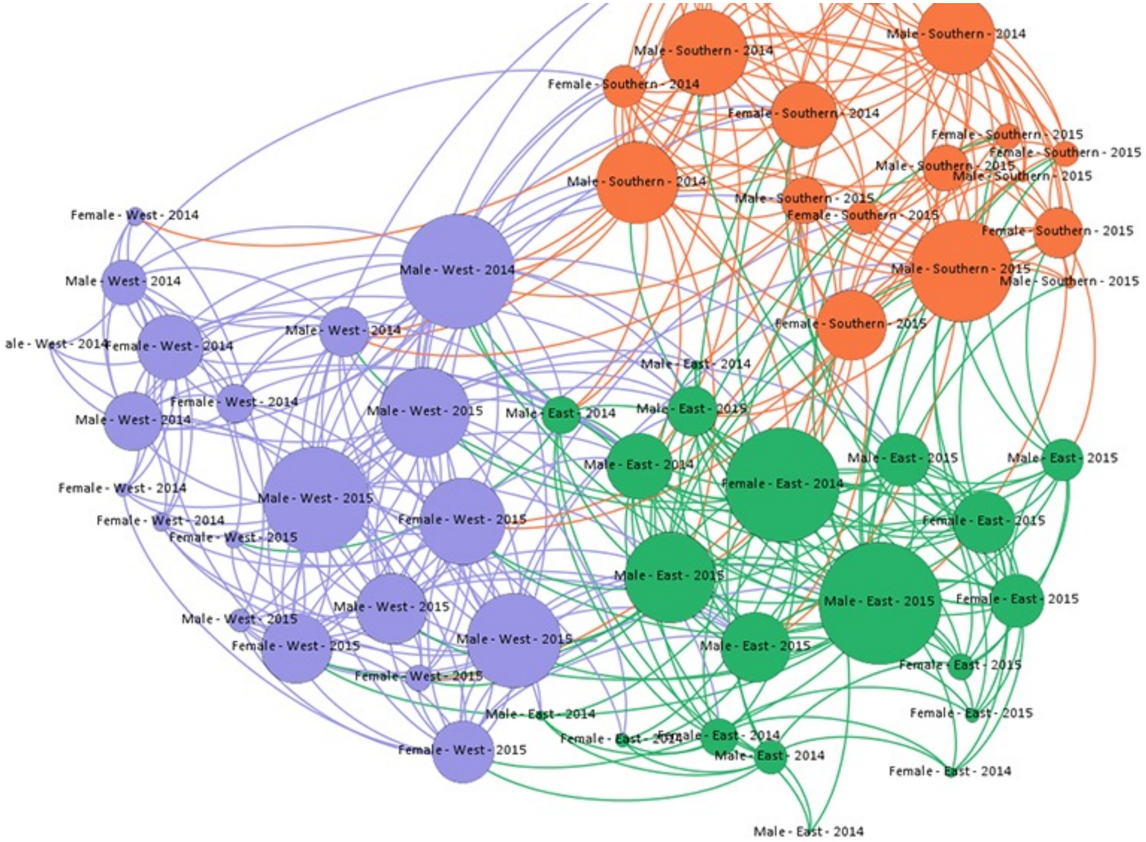
Arya: Arbitrary Graph Pattern Mining with Decomposition-based Sampling

Zeying Zhu*, Kan Wu*, Alan Zaoxing Liu

BOSTON UNIVERSITY

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

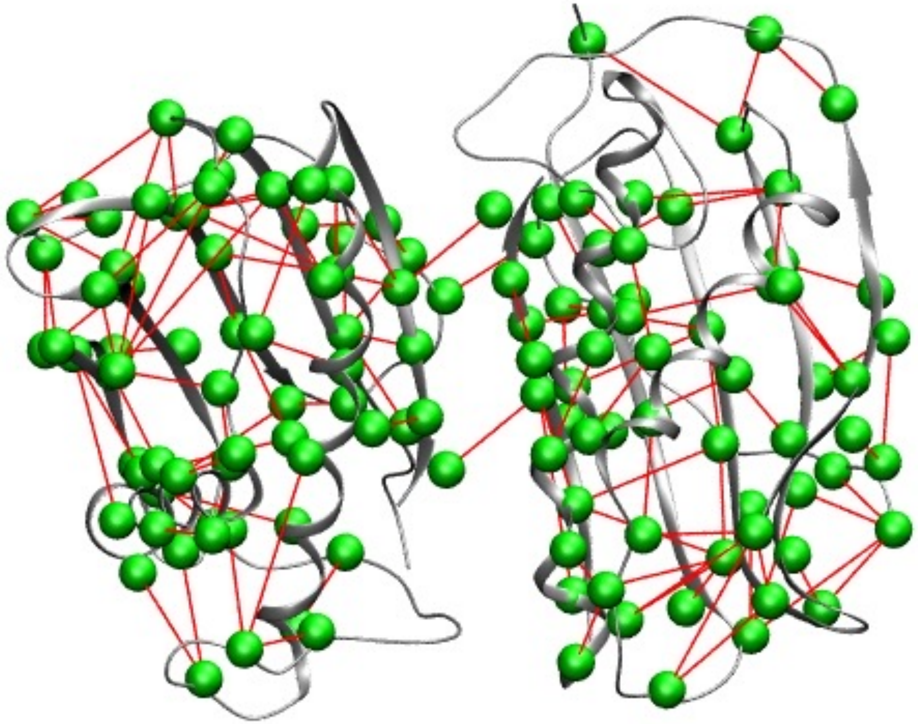
Graph-structured Data are Ubiquitous



Social networks

Twitter graph: ten billion edges*

*GraphJet: Real-Time Content Recommendations at Twitter

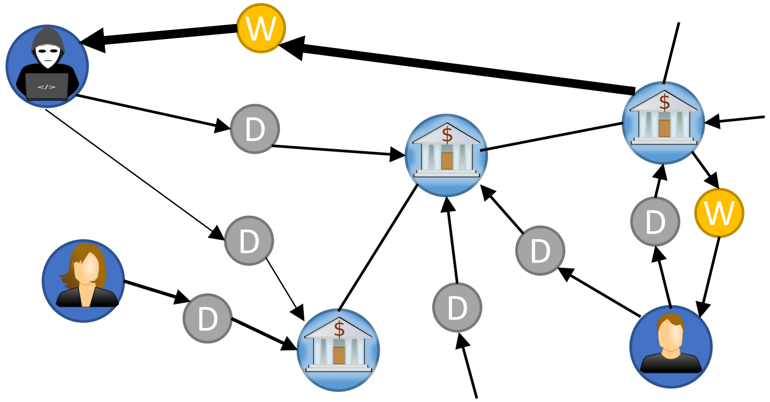


Protein-protein graph

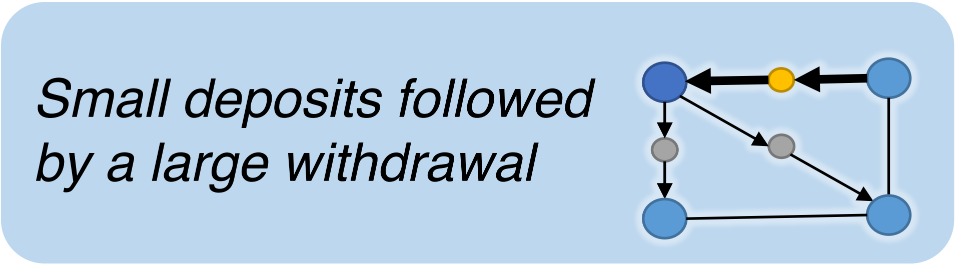
~80 billion nodes and 250 million edges

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

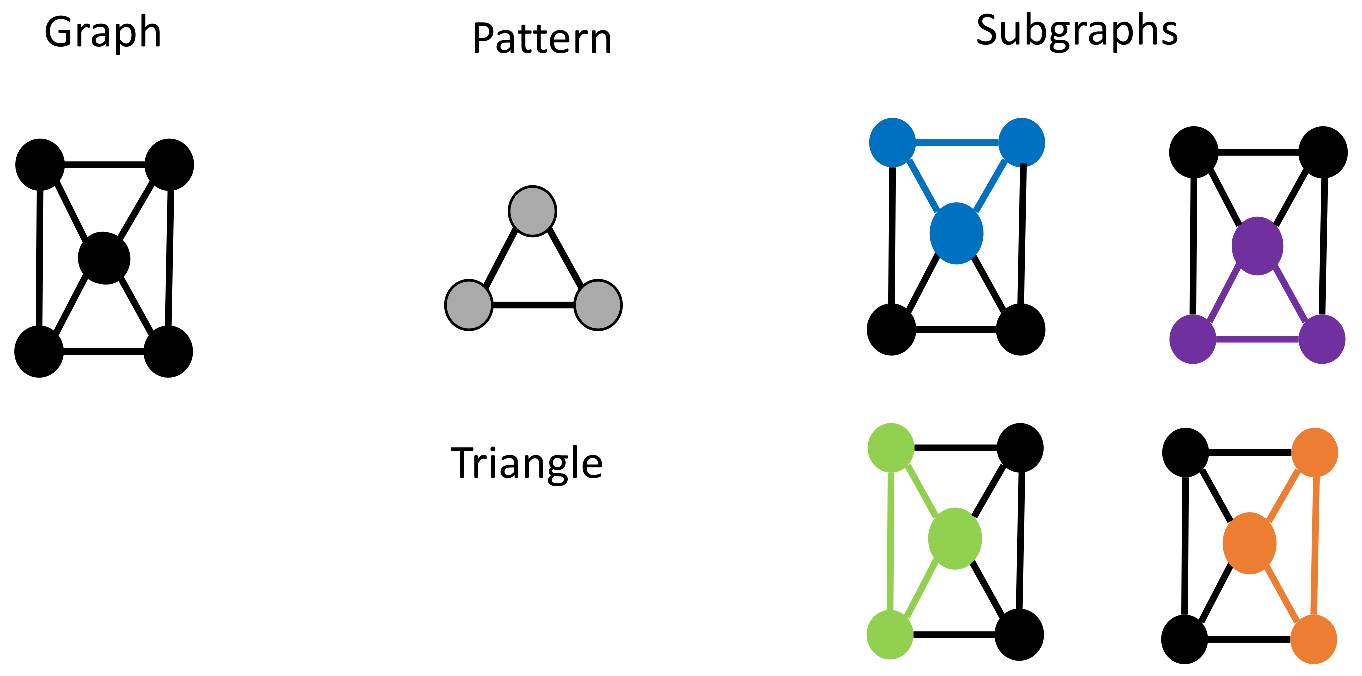


A simple pattern example



Graph Pattern Mining

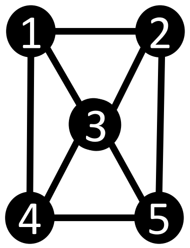
Find all subgraph instances matching a given pattern of interest.



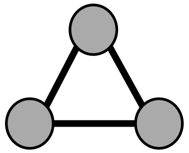
Counting the number of any subgraphs

Exact Mining Solutions

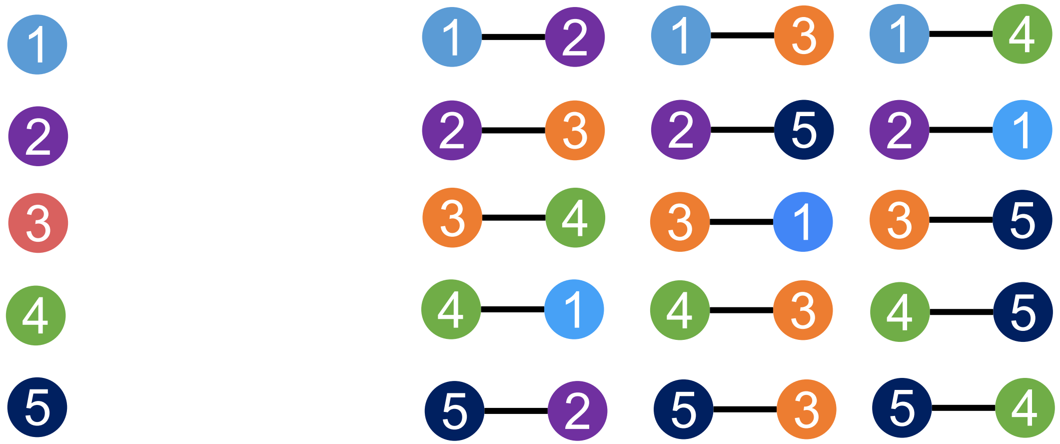
Graph



Pattern

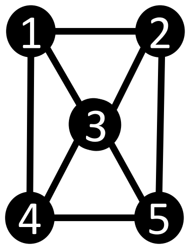


Iterate every isomorphic subgraph

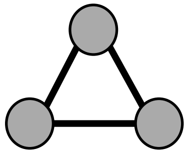


Exact Mining Solutions

Graph



Pattern

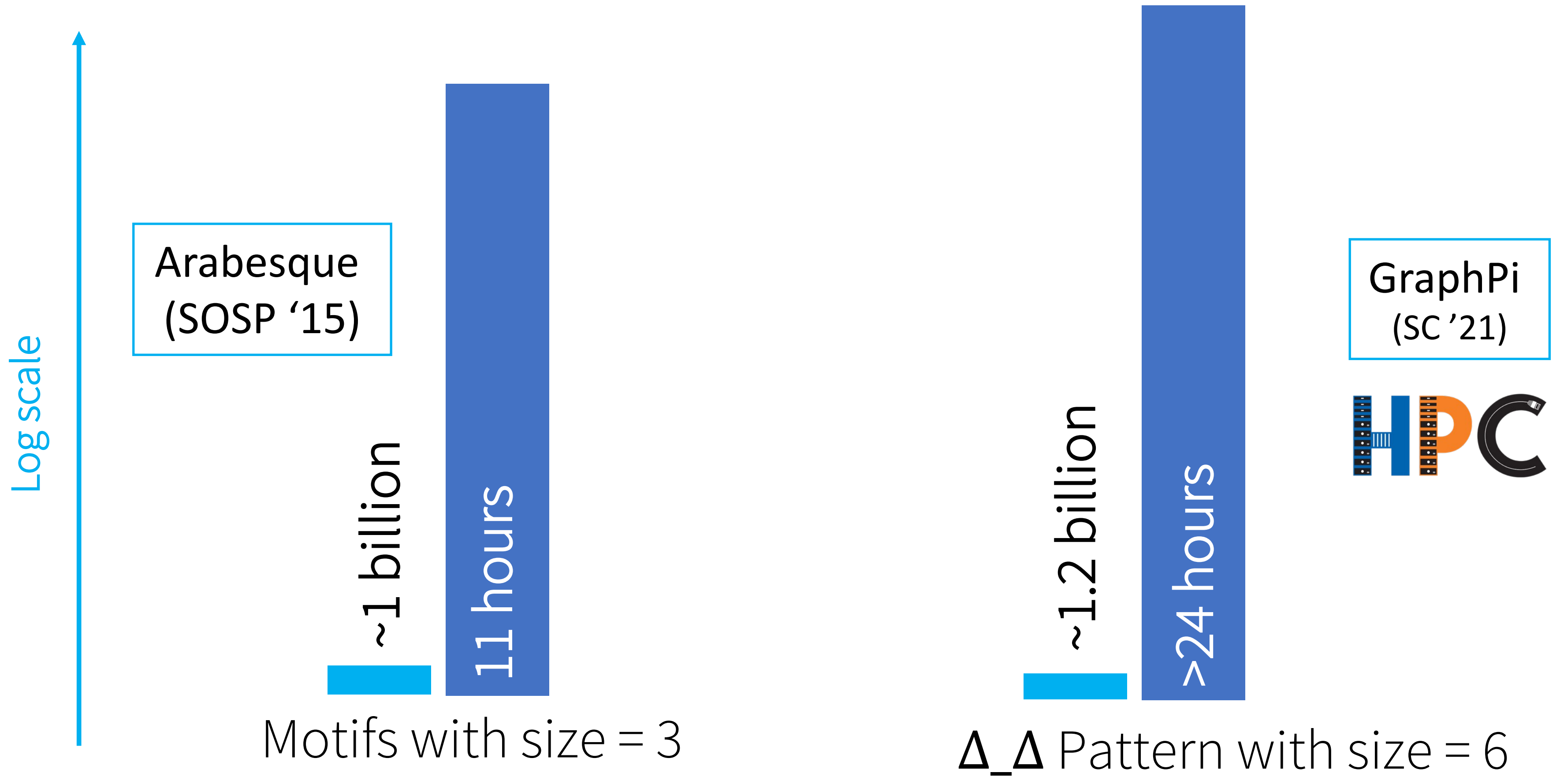


Iterate every isomorphic subgraph

Exponentially growing intermediate candidate sets

Optimizations: reduce redundant enumeration,
system optimizations, hardware accelerators,
but still **NP-Complete**

Scalability Challenge in Exact Mining



Approximate Pattern Mining

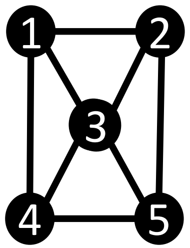
- Many mining tasks do not need exact answers.
  - Output density of certain patterns
- List some but not all subgraphs for large graphs.
  - Output representative ones

General approximate approach:
Sample a subset of the input data and estimate the count based on the probability.

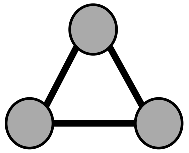
Using Neighborhood Sampling [ASAP, OSDI'18]

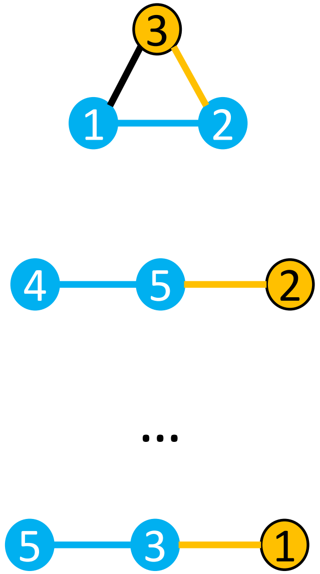
Sample a subset of the input data and estimate count based on probability

Graph



Pattern





$$p_1 = \frac{1}{8} \times \frac{1}{4} = \frac{1}{32} \qquad e_1 = 32$$

$$p_2 = \frac{1}{8} \times \frac{1}{4} = \frac{1}{32} \qquad e_2 = 0$$

… …

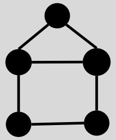
$$p_n = \frac{1}{8} \times \frac{1}{4} = \frac{1}{32} \qquad e_n = 0$$

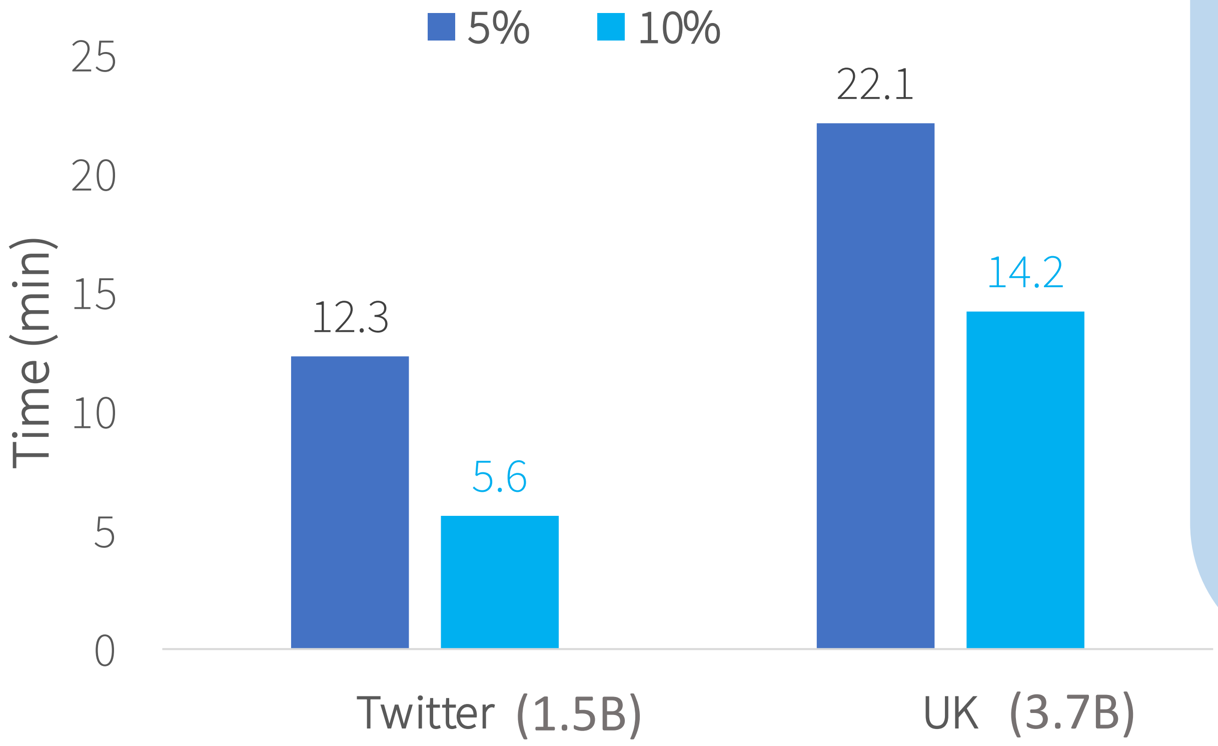
$$\frac{1}{n}\sum_{i=1}^{n} e_i \approx 4$$

Neighborhood sampling | Probability | △ Counting | Result

- Up to 258x speed up in ASAP

ASAP Cannot Scale to Complex Patterns

5-House 



16-machine cluster

- Need larger number of samplers for more complex patterns.
    - From 4-node to 5-node patterns, there is a O(Δ) increase.
    - Δ is the maximum degree of the graph.

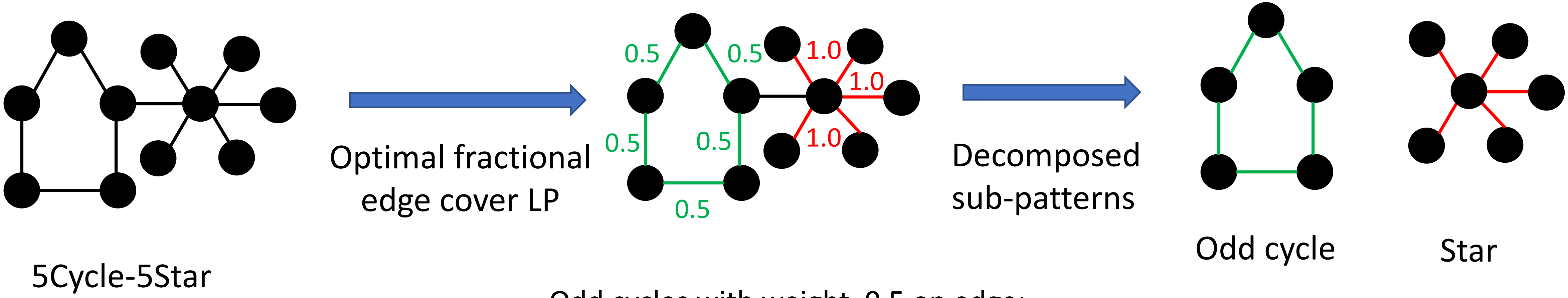
If the patterns are more complex, it needs significantly more samplers and is less scalable.

Can we reduce the complexity of the sampled pattern?

Our key idea is to leverage graph decomposition theory and sample different sub-patterns individually.

Graph Decomposition Theory

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.

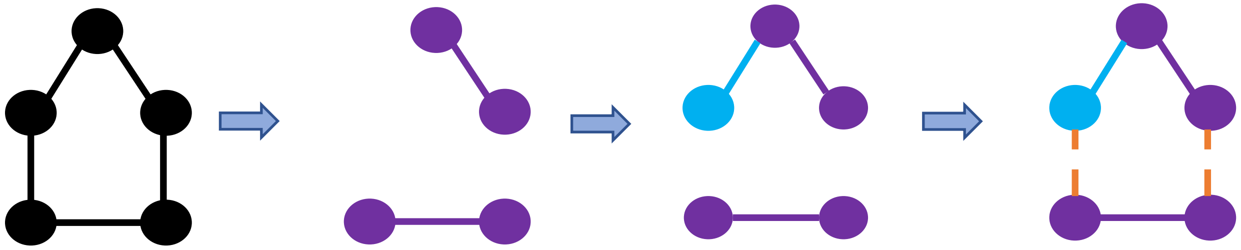


Odd cycles with weight 0.5 on edge;
Stars with weight 1 on each edge

Sample Individual Sub-patterns

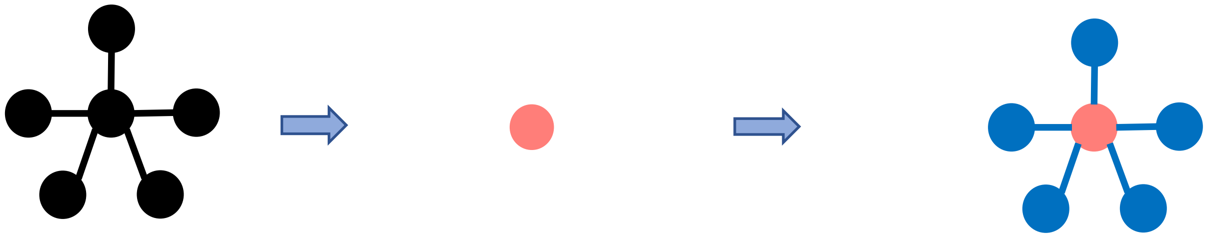
- Odd cycle sampler: edge sampling

5-Cycle



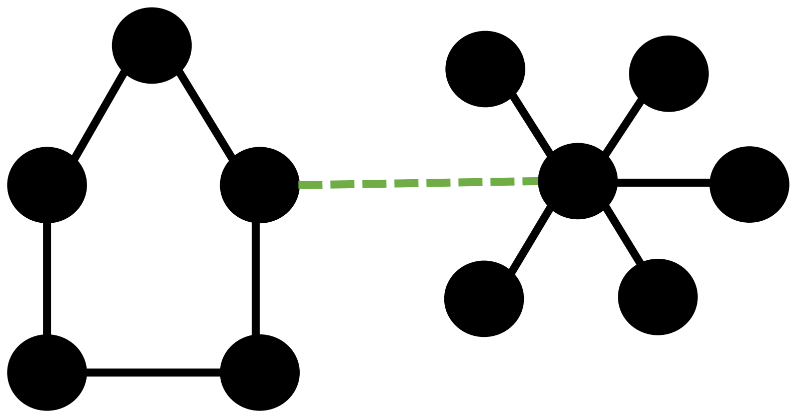
- Star sampler

5-Star



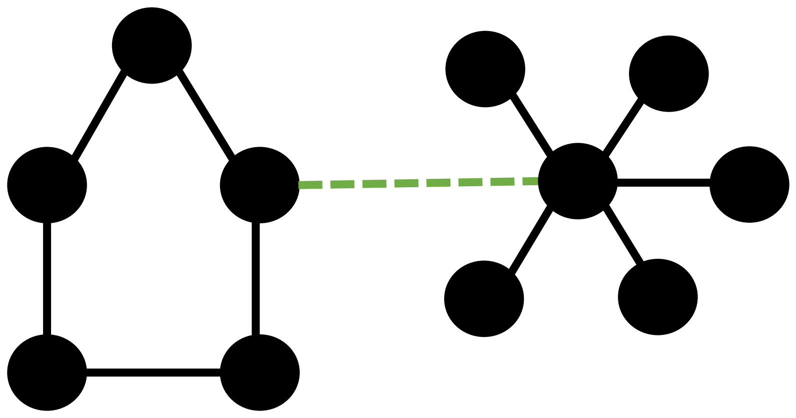
Form a Pattern

Test remaining edges in 5Cycle-5Star.



Accelerate the computation

Test remaining edges in 5Cycle-5Star.



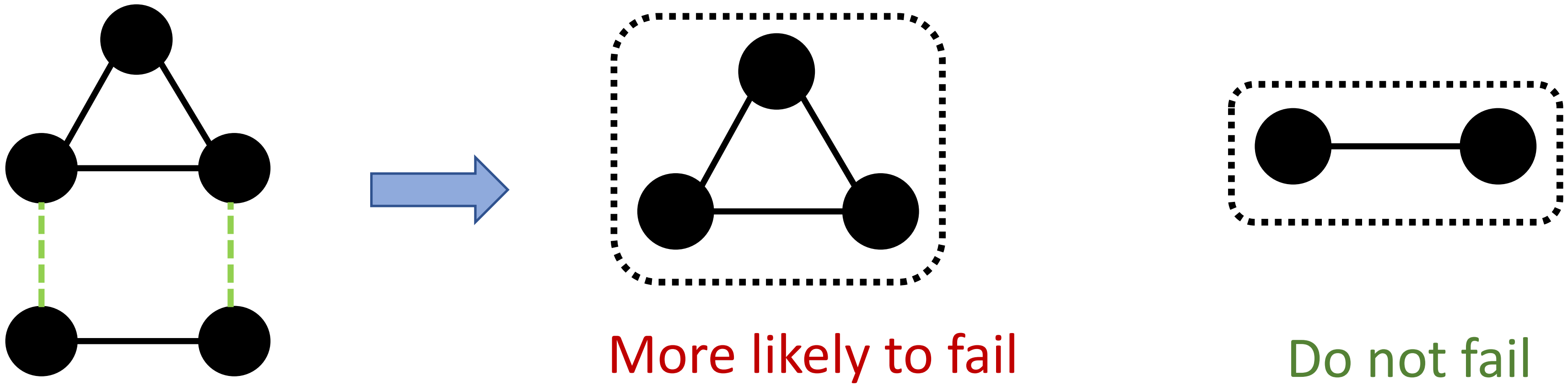
System optimizations:

- **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

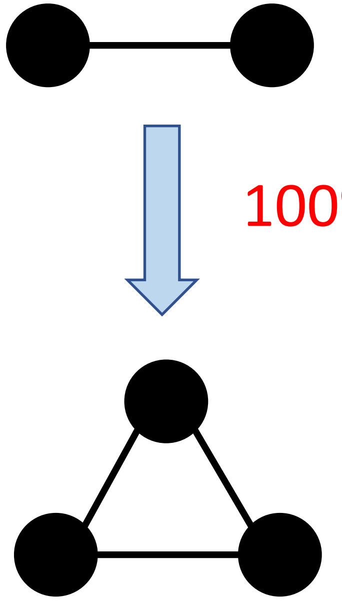


More likely to fail
e.g., 92%

Do not fail

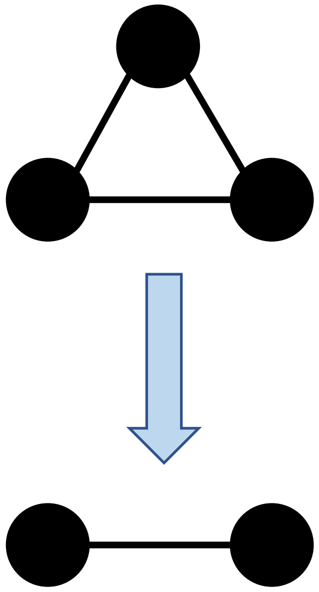
- The subpattern sampling order matters for mining time!
  - If any subpattern sampling fails, the entire pattern sampling fails.
  - We can early terminate the sampling if there are any failures.

Scheduling More-likely-to-fail Subpatterns First



0% fail

100% sample triangle

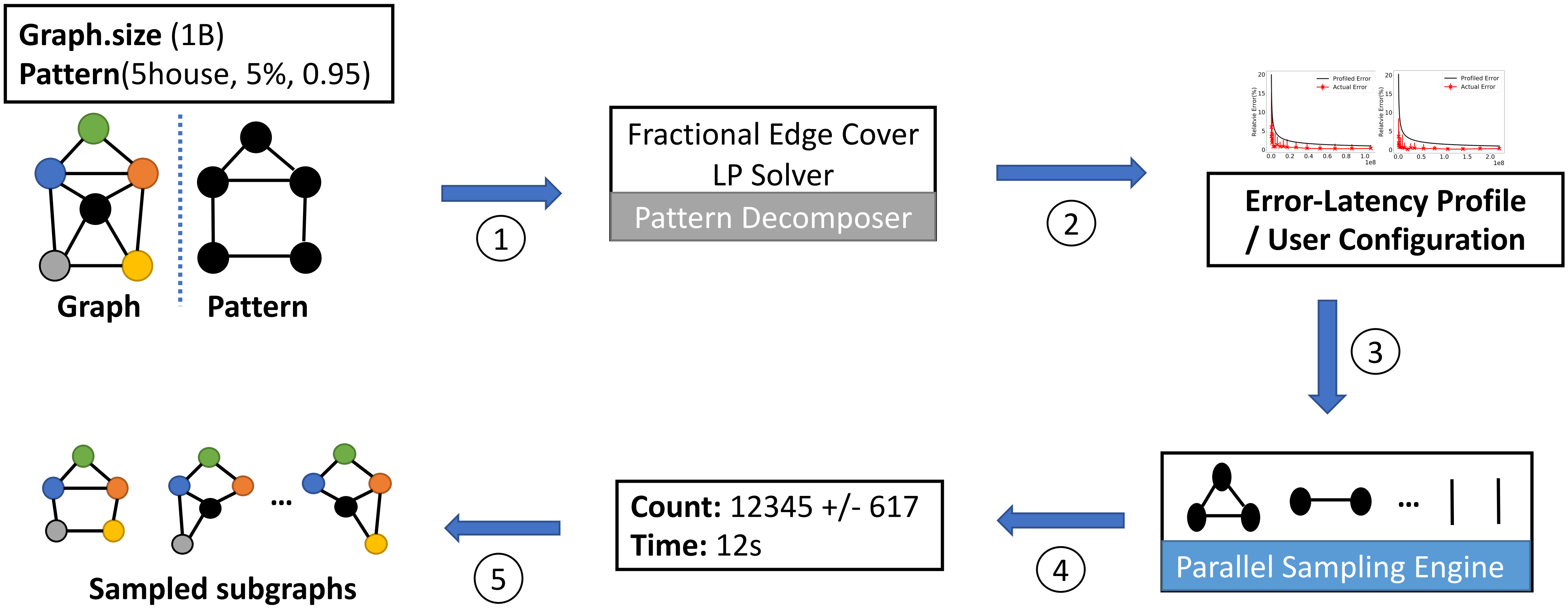


92% fail

8% sample 1-star

Improve the performance by 2x without affecting accuracy.

Putting It Together -- Arya

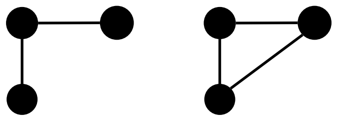


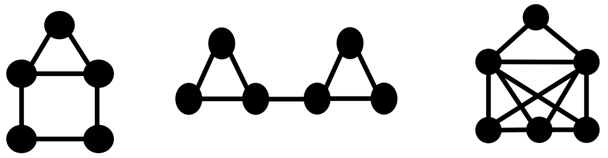
Evaluation

- Distributed system implementation (11K LOC) **<5% error**
  - 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

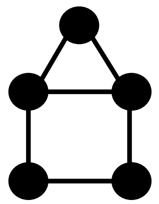
- Patterns
  - 3-Motifs (2 patterns), 4-Motifs (6 patterns), complex patterns (>= 5 nodes)

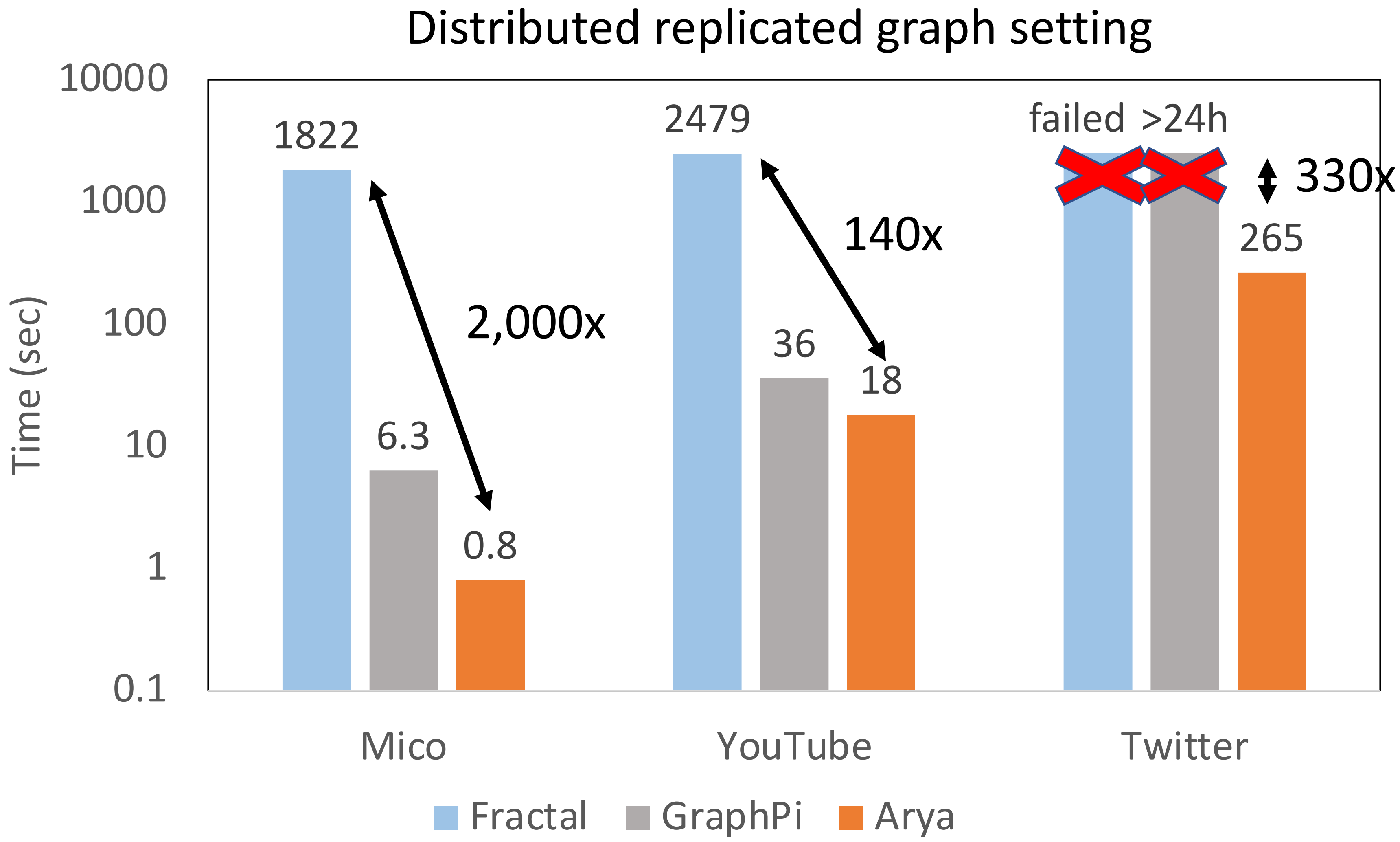




Evaluation: Exact Mining Systems

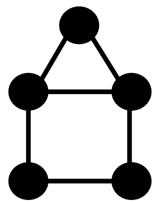
5-House

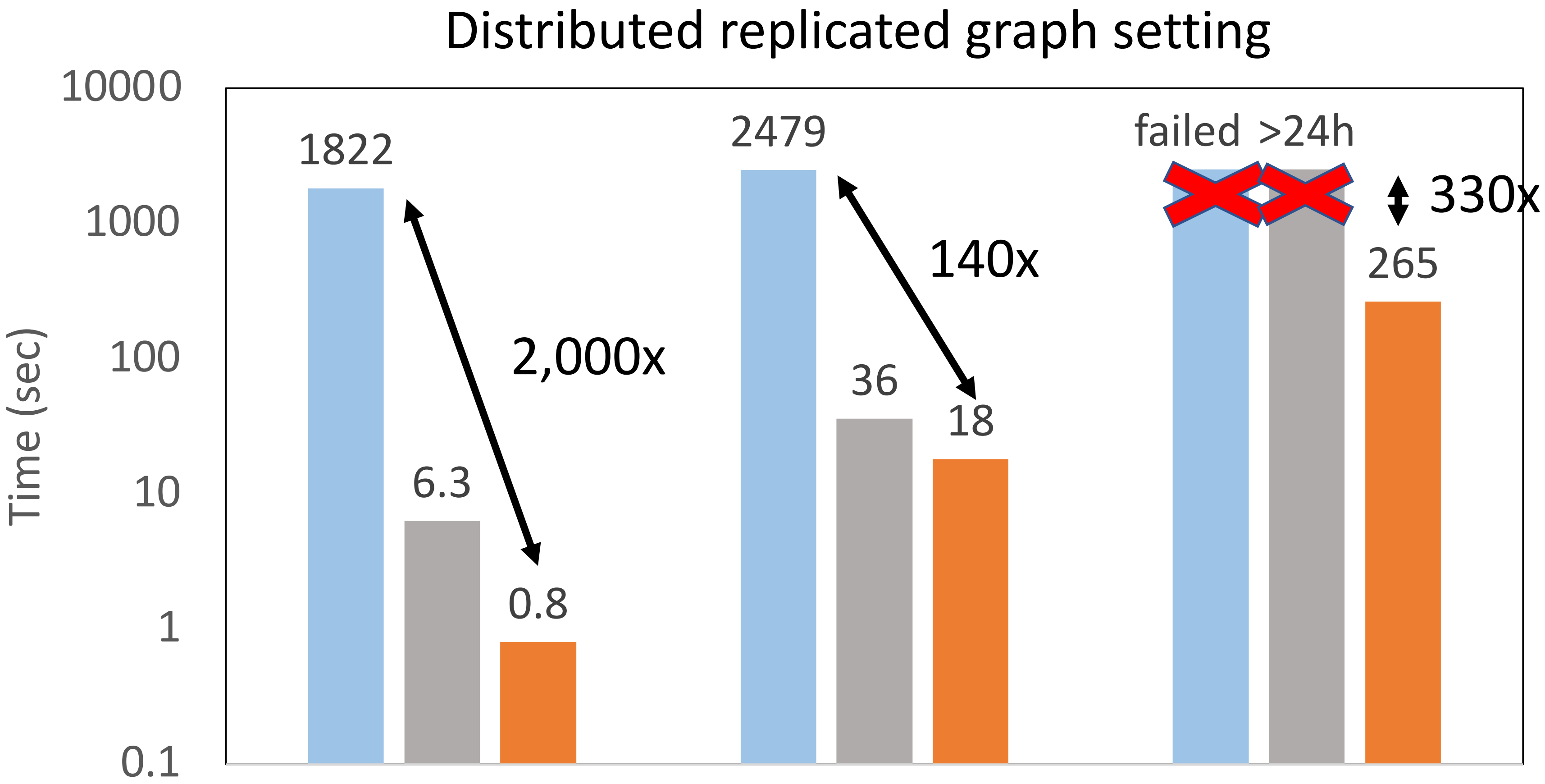




Evaluation: Exact Mining Systems

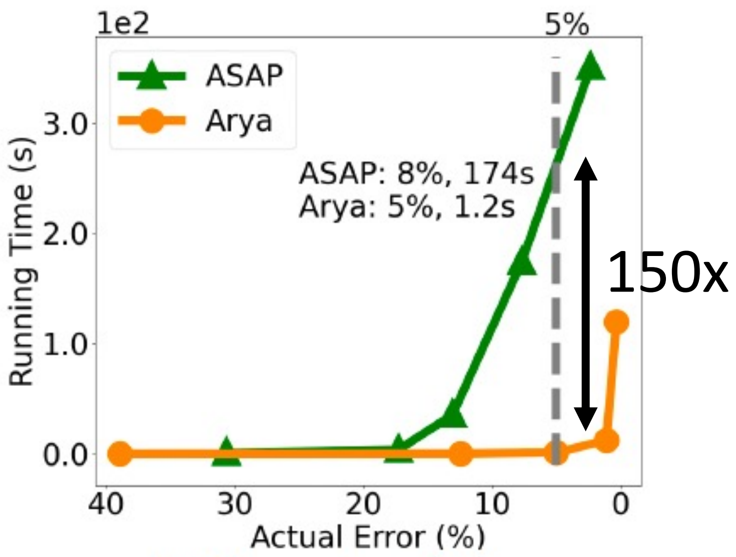
5-House



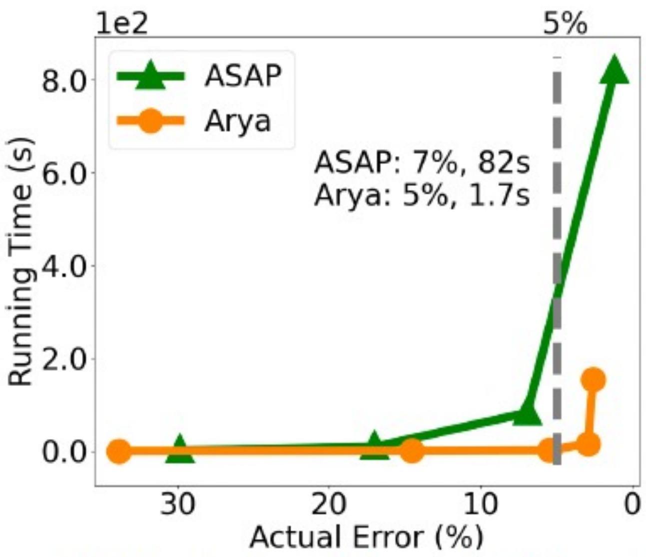


Up to 20,000x faster than Fractal
Up to 1,000x faster than GraphPi

Evaluation: Approximate Mining Systems



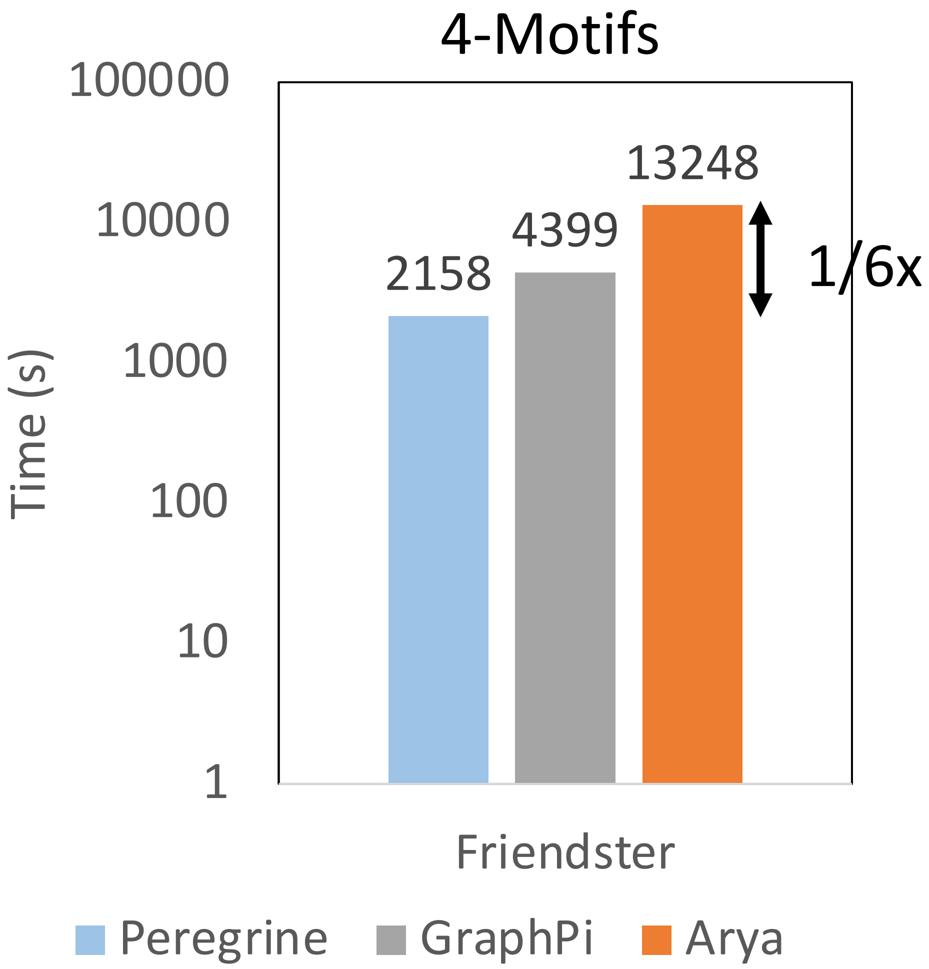
YouTube, 5-House



YouTube, Triangle-Triangle

- 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.