COL761 HW1: Frequent Subgraph Mining Analysis

Arnav Raj [2022CS51652]
Arpit Agrawal [2022CS11612]
Ayush Gupta [2022CS11114]

February 10, 2025

Abstract

This report analyzes the performance of three frequent subgraph mining algorithms—gSpan, FSG (PAFI), and Gaston—using the Yeast dataset. We evaluate their execution times at various support thresholds (5%, 10%, 25%, 50%, and 95%) and discuss the observed trends and trade-offs inherent to each algorithm.

1 Introduction

Frequent subgraph mining is essential for uncovering recurring patterns in graph-structured data, with applications in bioinformatics, cheminformatics, and social network analysis. In this study, we focus on three algorithms:

- **gSpan**: Uses depth-first search (DFS) combined with canonical labeling to enumerate frequent subgraphs.
- **FSG (PAFI)**: Employs breadth-first search (BFS) with aggressive candidate pruning to efficiently mine subgraphs.
- **Gaston**: Utilizes pattern growth and embedding-based optimizations to handle the computational challenges of subgraph isomorphism.

We test these algorithms on the Yeast dataset over a range of support thresholds to assess their scalability and performance.

2 Methodology

The experiments were conducted using a shell script (`q2.sh`) that accepts the paths to the algorithm executables, the Yeast dataset file, and an output directory. Our setup includes:

- **Dataset**: The Yeast dataset, formatted as per the assignment guidelines.
- **Support Thresholds**: Experiments were performed at 5%, 10%, 25%, 50%, and 95% to capture a broad range of conditions.
- **Execution**: Runtime metrics and output files are collected for each algorithm and support level, enabling direct performance comparison.

3 Results and Discussion

Figure 1 illustrates the running times of gSpan, FSG, and Gaston at different minimum support thresholds.

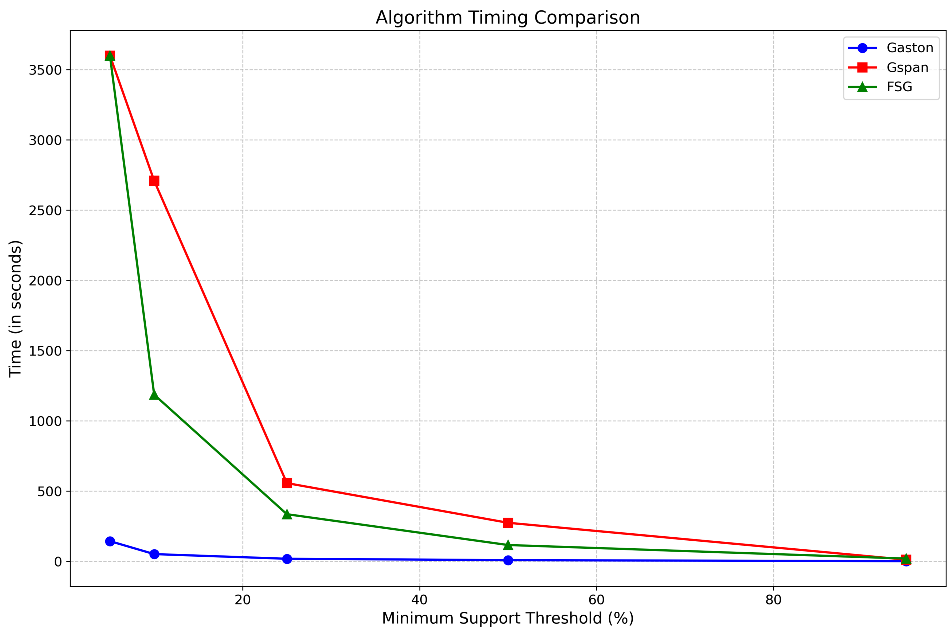

Figure 1: Execution Time vs. Support Threshold for gSpan, FSG, and Gaston.

3.1 Analysis

gSpan:

- gSpan shows an exponential increase in runtime as the support threshold decreases, with the steepest rise at 5%.
- Its DFS-based search and canonical labeling, while thorough, incur high computational costs for low-support scenarios.

FSG (PAFI):

- FSG benefits from effective candidate pruning via a BFS approach, resulting in better performance than gSpan across most thresholds.
- Nevertheless, at very low supports, the large number of candidate subgraphs still causes significant runtime increases.

Gaston:

- Gaston consistently exhibits the lowest execution times, even at low support thresholds, due to its efficient pattern growth strategy and embedding optimizations.
- This robust performance makes Gaston the most scalable option for dense graph datasets.

Overall Observations:

- Lowering the support threshold increases the number of frequent subgraphs, thereby raising the computational demand for all algorithms.
- Gaston's performance suggests it is best suited for large-scale subgraph mining tasks, whereas gSpan and FSG may be more appropriate when moderate support thresholds are acceptable.

4 Conclusion

Our analysis shows that while all three algorithms face increased runtimes at low support thresholds, Gaston is the most efficient and scalable. gSpan, despite its comprehensive search method, and FSG, with its candidate pruning, both suffer from performance bottlenecks when the dataset is dense. These results can guide the selection of appropriate subgraph mining techniques based on the specific requirements and constraints of the application.

5 References

1. Yan, X., & Han, J. (2002). *gSpan: Graph-based Substructure Pattern Mining.* In Proceedings of the IEEE International Conference on Data Mining.

2. Papadimitriou, S., et al. (2003). *Frequent Subgraph Mining.* ACM SIGKDD Explorations Newsletter.

3. Gaston Library Documentation. Retrieved from `https://liacs.leidenuniv.nl/~nijssensgr/gaston/download.html`