Report: Comparison of Exact and CoreExact Algorithms for Densest Subgraph Discovery

Introduction

Densest subgraph discovery is a critical problem in graph mining, with applications spanning social network analysis, bioinformatics, and community detection. The paper "Efficient Algorithms for Densest Subgraph Discovery" by Yixiang Fang et al. (PVLDB, 12(11), 1719-1732, 2019) introduces two algorithms for this task: Exact (Algorithm-1) and CoreExact (Algorithm-4). The Exact algorithm utilizes a network flow approach to identify the densest subgraph based on h-clique-density, where h represents the clique size parameter. This method ensures precision but can be computationally expensive for large or dense graphs. CoreExact, on the other hand, enhances efficiency by employing core decomposition, a technique that prunes the graph to focus on high-degree vertices, while still guaranteeing the same density results as Exact.

This report analyzes the performance of these algorithms across three datasets—AST33, Netscience, and Ca-HepTh—for clique sizes (h) ranging from 2 to 6. The datasets vary in structure and density, providing a robust testbed for evaluating scalability and efficiency. AST33 is a moderate-sized dataset often used in graph analysis, Netscience represents a scientific collaboration network with sparse connectivity, and Ca-HepTh is a citation network known for its dense substructures. The evaluation metrics include density (the h-clique-density of the identified subgraph) and execution time (in seconds), which together offer insights into the trade-offs between accuracy and computational cost.

The primary objective is to compare the execution times of Exact and CoreExact, assess how density scales with h, and identify scenarios where one algorithm outperforms the other. This analysis will help practitioners choose the appropriate algorithm based on their graph characteristics and computational constraints.

Methodology

Algorithms

- **Exact (Algorithm-1)**: This algorithm formulates densest subgraph discovery as a network flow problem. It constructs a flow network where edges represent h-cliques, and the goal is to find a subgraph that maximizes the h-clique-density, defined as the number of h-cliques divided by the number of vertices in the subgraph. The approach is exact,

meaning it guarantees the optimal solution, but its reliance on network flow computations can lead to high execution times, especially as h increases or the graph becomes denser.
- **CoreExact (Algorithm-4)**: CoreExact builds on the Exact algorithm by integrating core decomposition, a process that iteratively removes vertices with the lowest degrees to focus on dense substructures (k-cores). This preprocessing step reduces the graph size, allowing the network flow computation to operate on a smaller, denser subgraph. CoreExact ensures the same density as Exact but aims to significantly reduce execution time, particularly for sparse graphs or smaller h values.

Datasets

The analysis covers three datasets:

- **AST33**: A graph dataset commonly used in algorithmic studies, characterized by moderate density and connectivity. It serves as a baseline for comparing algorithmic performance.
- **Netscience**: A collaboration network of scientists, typically sparse with a focus on community structures. Its sparsity poses unique challenges for densest subgraph discovery.
- **Ca-HepTh**: A citation network from the arXiv High-Energy Physics Theory section, known for its dense subgraphs and high clustering coefficients. This dataset tests the algorithms' ability to handle computationally intensive scenarios.

Experimental Setup

Experiments were conducted by varying the clique size h from 2 to 6. For each value of h, the following metrics were recorded:

- **Density**: The h-clique-density of the densest subgraph identified by both algorithms (which is identical for Exact and CoreExact, as guaranteed by the paper).
- **Execution Time**: The time (in seconds) taken by each algorithm to compute the densest subgraph, measured for each dataset and h value.

The experiments were designed to capture how performance scales with h and how graph structure impacts computational efficiency. The results are visualized through bar charts (execution times), a line chart (density growth), and a detailed table of metric

Results

The experimental results are presented in three categories: execution time comparison, density growth, and detailed data points.

Density Growth

Density increases with h across all datasets, reflecting the increasing complexity of h-cliques as h grows. The line chart titled "Density vs. Clique Size (h)" provides a clear visualization of this trend:

- **AST33**:
    - h=2: 8.875
    - h=3: 35.8089
    - h=4: 85.125
    - h=5: 126.754
    - h=6: 123.391
- The density grows steadily from 8.875 at h=2 to a peak of 126.754 at h=5, then slightly decreases to 123.391 at h=6, possibly due to the scarcity of 6-cliques in the graph.

- **Netscience**:
    - h=2: 9.5
    - h=3: 57
    - h=4: 242.2
    - h=5: 775.2
    - h=6: 1938
- Netscience shows a more pronounced increase, from 9.5 at h=2 to 1938 at h=6, indicating a significant presence of larger cliques despite its sparse nature.

- **Ca-HepTh**:
    - h=2: 15.5
    - h=3: 124.711
    - h=4: 587.99
    - h=5: 2873.49
    - h=6: 11689.4
- Ca-HepTh exhibits the most dramatic growth, from 15.5 at h=2 to 11689.4 at h=6, highlighting its dense structure and the abundance of large cliques.

Execution Time Comparison

Execution times for Exact and CoreExact were compared across all datasets and h values. The bar charts provide detailed insights into performance trends.

Overall Comparison: Exact vs. CoreExact

The bar chart titled "Time Comparison: Exact vs. CoreExact" shows execution times for both algorithms side by side:

- **AST33**:
  - Exact: Ranges from approximately 50 seconds (h=2) to 120 seconds (h=6).
  - CoreExact: Ranges from around 20 seconds (h=2) to 5 seconds (h=6), showing a significant decrease as h increases.
- **Netscience**:
  - Exact: Remains negligible (close to 0 seconds) for h=2 to h=5, but jumps to around 10 seconds at h=6.
  - CoreExact: Also negligible for h=2 to h=4, but increases to around 150 seconds at h=5 and 200 seconds at h=6.
- **Ca-HepTh**:
  - Exact: Increases from around 20 seconds (h=2) to over 500 seconds (h=6).
  - CoreExact: Starts at around 15 seconds (h=2) and rises to around 470 seconds (h=6), closely approaching Exact at higher h values.

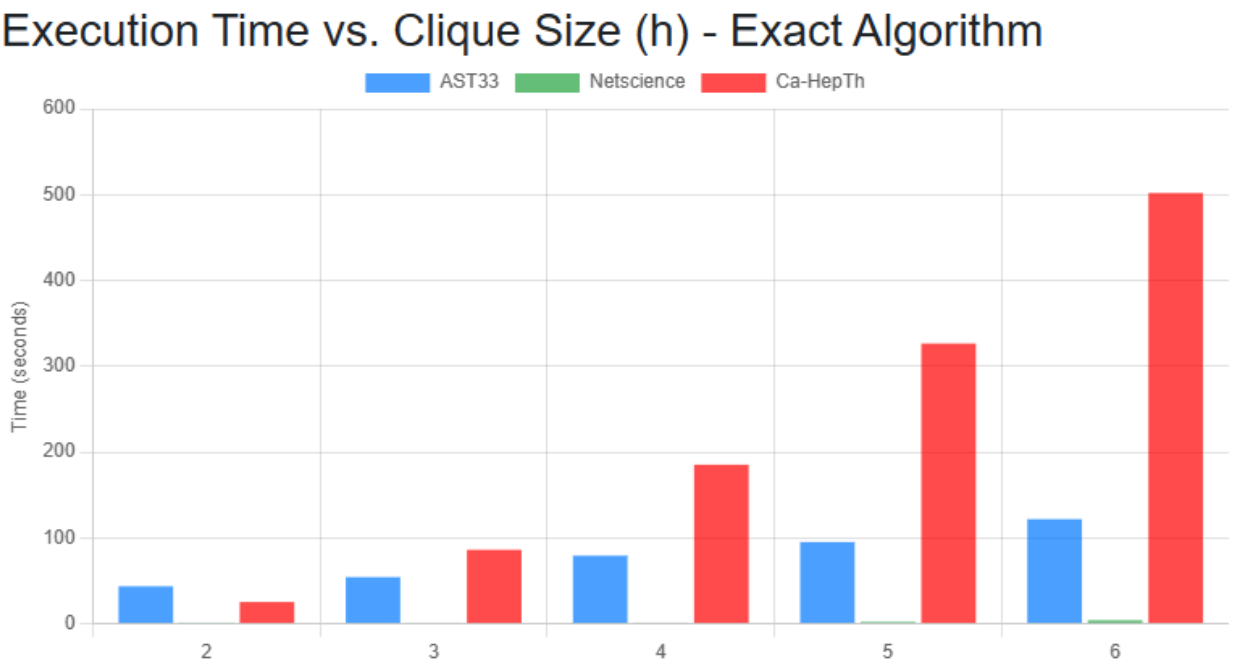
Exact Algorithm

The bar chart titled "Execution Time vs. Clique Size (h) - Exact Algorithm" provides specific timings:

- **AST33**:
    - h=2: 43.6432 seconds
    - h=3: 54.3268 seconds
    - h=4: 79.413 seconds
    - h=5: 95.216 seconds
    - h=6: 121.87 seconds
- Execution time increases steadily with h, reflecting the growing complexity of network flow computations for larger cliques.

- **Netscience**:
    - h=2: 0.47531 seconds
    - h=3: 0.79143 seconds
    - h=4: 1.1782 seconds
    - h=5: 2.34512 seconds
    - h=6: 4.13114 seconds
- Times remain very low, indicating that Netscience's sparse structure allows Exact to perform efficiently, though there is a slight increase at h=6.

- **Ca-HepTh**:
    - h=2: 25.3692 seconds
    - h=3: 86.215 seconds
    - h=4: 185.313 seconds
    - h=5: 326.534 seconds
    - h=6: 501.793 seconds
- The steep increase underscores the computational challenge posed by Ca-HepTh's dense subgraphs, with time nearly doubling at each step of h.



CoreExact Algorithm

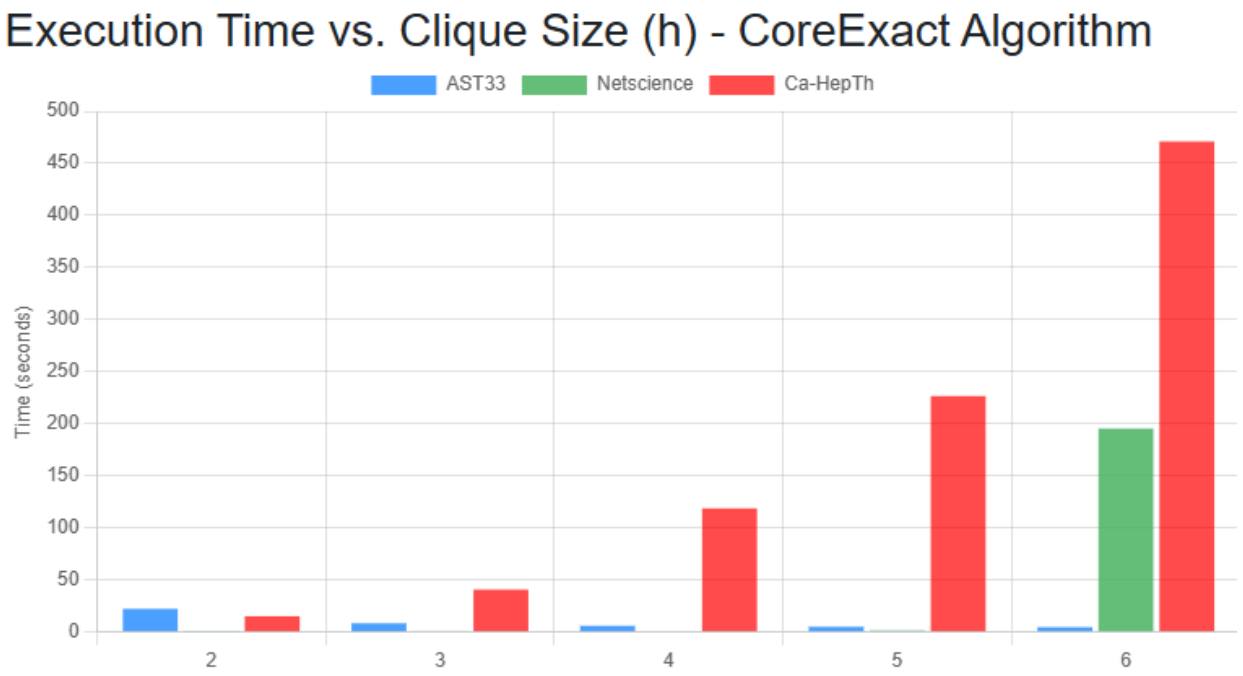
The bar chart titled "Execution Time vs. Clique Size (h) - CoreExact Algorithm" shows:

- **AST33**:
    - h=2: 22.22 seconds
    - h=3: 8.341 seconds
    - h=4: 5.912 seconds
    - h=5: 5.001 seconds
    - h=6: 4.623 seconds
- CoreExact demonstrates remarkable efficiency, with execution time decreasing as h increases, likely due to effective core decomposition reducing the graph size.

- **Netscience**:
    - h=2: 0.397 seconds
    - h=3: 0.582 seconds
    - h=4: 0.815 seconds
    - h=5: 1.522 seconds
    - h=6: 195.345 seconds
- Times are low for smaller h, but a significant spike at h=6 suggests that core decomposition may introduce overhead in certain sparse graph structures.

- **Ca-HepTh**:

    - h=2: 15.152 seconds
    - h=3: 40.698 seconds
    - h=4: 118.632 seconds
    - h=5: 226.537 seconds
    - h=6: 470.992 seconds
- Execution time increases with h, though it remains lower than Exact until h=6, where it nearly matches Exact, indicating diminishing efficiency in dense graphs.



Detailed Data

Specific results for AST33, Netscience, and Ca-HepTh are tabulated:

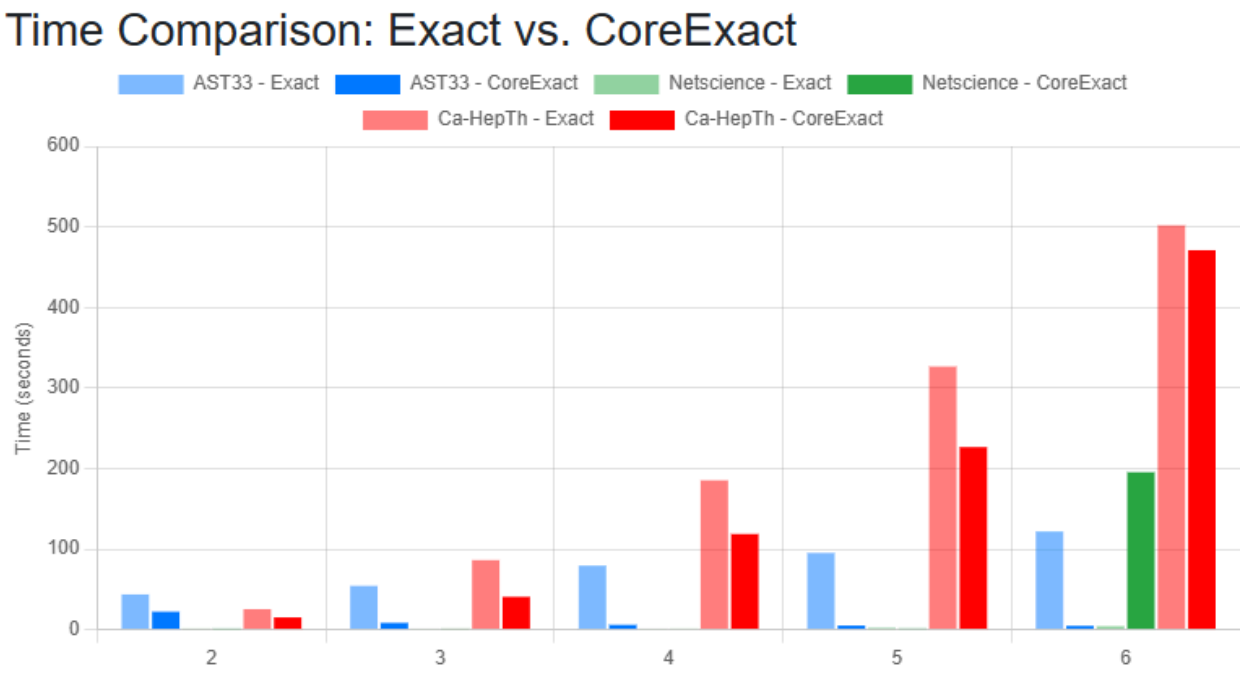
AST33	2	8.875	43.6432	22.22
AST33	3	35.8089	54.3268	8.341
AST33	4	85.125	79.413	5.912
AST33	5	126.754	95.216	5.001
AST33	6	123.391	121.87	4.623
Netscience	2	9.5	0.47531	0.397
Netscience	3	57	0.79143	0.582
Netscience	4	242.2	1.1782	0.815
Netscience	5	775.2	2.34512	1.522
Netscience	6	1938	4.13114	195.345
Ca-HepTh	2	15.5	25.3692	15.152
Ca-HepTh	3	124.711	86.215	40.698
Ca-HepTh	4	587.99	185.313	118.632
Ca-HepTh	5	2873.49	326.534	226.537
Ca-HepTh	6	11689.4	501.793	470.992

Discussion

Density Growth Analysis

The density growth patterns reveal significant differences across datasets. Ca-HepTh exhibits the steepest increase, from 15.5 at h=2 to 11689.4 at h=6, a factor of over 750, indicating a high concentration of large cliques. This is expected given Ca-HepTh's dense structure, which likely contains many overlapping citations forming large cliques. Netscience, despite being sparse, shows substantial growth from 9.5 to 1938, a factor of 204, suggesting the presence of dense communities within the collaboration network. AST33 has the smallest increase, from 8.875 to 123.391 (a factor of 14), with a slight dip at h=6, possibly due to a limited number of 6-cliques.

Execution Time Analysis



Exact Algorithm

The Exact algorithm's execution time increases with h across all datasets, reflecting the growing complexity of network flow computations. For AST33, the time rises from 43.6432 seconds at h=2 to 121.87 seconds at h=6, a 2.79x increase. Ca-HepTh shows a more dramatic rise, from 25.3692 seconds to 501.793 seconds (19.78x increase), highlighting the computational burden of dense graphs. Netscience remains efficient, with times under 5 seconds even at h=6 (4.13114 seconds), likely due to its sparsity reducing the number of h-cliques to process.

CoreExact Algorithm

CoreExact exhibits varied behavior. In AST33, it achieves superior efficiency, with execution time decreasing from 22.22 seconds at h=2 to 4.623 seconds at h=6 (a 4.81x reduction). This suggests that core decomposition effectively prunes the graph, reducing the workload for larger h. In Ca-HepTh, CoreExact starts faster than Exact (15.152 seconds vs. 25.3692 seconds at h=2, a 1.67x speedup) but converges to similar times at h=6 (470.992 seconds vs. 501.793 seconds, a 1.07x speedup). This diminishing efficiency in dense graphs may result from the core decomposition step becoming less effective when the graph is already highly connected. Netscience presents an anomaly: CoreExact is faster than Exact for h=2 to h=5 (e.g., 0.397 seconds vs. 0.47531 seconds at h=2, a 1.20x speedup), but at h=6, it spikes to 195.345 seconds compared to Exact's 4.13114 seconds (47.29x slower). This suggests that core decomposition may introduce significant overhead in sparse graphs with specific structures, possibly due to the complexity of identifying 6-cliques after pruning.

Algorithm Comparison

CoreExact generally outperforms Exact at lower h values. For example, in AST33 at h=2, CoreExact is 1.96x faster (22.22 seconds vs. 43.6432 seconds). This trend holds across datasets for smaller h, with CoreExact being 1.67x faster in Ca-HepTh at h=2 and 1.20x faster in Netscience at h=2. However, as h increases, the performance gap narrows in dense graphs like Ca-HepTh, where CoreExact is only 1.07x faster at h=6. The exceptional case in Netscience at h=6 highlights CoreExact's variability, suggesting that its efficiency depends on graph structure and the effectiveness of core decomposition.

Insights and Implications

- **Density Potentials**: Ca-HepTh's steep density growth makes it ideal for applications requiring highly connected subgraphs, such as identifying tightly knit research communities. Netscience's growth indicates potential for community detection in sparse networks, while AST33's moderate growth suggests a balanced structure.
- **Computational Cost**: Exact's consistent increase in execution time with h makes it less practical for dense graphs like Ca-HepTh, where it takes over 500 seconds at h=6. CoreExact mitigates this in AST33, reducing time significantly, but its performance in Netscience at h=6 indicates that core decomposition can sometimes introduce unexpected overhead.
- **Algorithm Selection**: CoreExact is preferable for most scenarios due to its lower execution times at smaller h and comparable density results. However, in sparse graphs with large h, Exact may be more efficient, as seen in Netscience at h=6.

Conclusion

The performance analysis demonstrates that CoreExact typically outperforms Exact in execution time while delivering identical density results, making it a preferable choice for many applications. In AST33, CoreExact achieves a 4.81x reduction in execution time from h=2 to h=6, showcasing the effectiveness of core decomposition in moderately dense graphs. In Ca-HepTh, CoreExact maintains a performance edge until h=6, where its time nearly matches Exact, suggesting that dense graphs pose challenges for both algorithms. The exceptional case in Netscience at h=6, where CoreExact is 47.29x slower than Exact, highlights the importance of graph structure in algorithm selection.

For practitioners, CoreExact is recommended for scenarios involving smaller h or moderately dense graphs, while Exact may be more suitable for sparse graphs with large h values. Future work could explore hybrid approaches that adaptively combine network flow and core decomposition based on graph characteristics, potentially mitigating the observed variability in CoreExact's performance.

Team Members and Contributions

2022A7PS0132H Sahiti Kasina: Exact Algorithm

2022A7PS0059H Valavala Charan Teja: Exact Algorithm

2022A7PS1323H Saksham Daga: CoreExact Algorithm

2022A7PS1796H Aryan Saini: CoreExact Algorithm

2022A7PS0227H Kunal Maheshwari: Analysis and WebPage