ASSIGNMENT 1

CS F364 - Design & Analysis of Algorithim

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
HYDERABAD CAMPUS

Under Supervision of
Prof. Apurba Das

Submitted by:

Name	ID	Email
Mehul Kochar	2021B3A73032H	f20213032@hyderabad.bits-pilani.ac.in
Arihant Rai	2021B3A72507H	f20212507@hyderabad.bits-pilani.ac.in
Kinjal Varida	2021B3A72579H	f20212579@hyderabad.bits-pilani.ac.in
Deepanshu Garg	2021B3A72758H	f20212758@hyderabad.bits-pilani.ac.in
Shubham Birla	2021B3A72965H	f20212965@hyderabad.bits-pilani.ac.in

Contents

CS F364: DAA

1 Introduction

Graph theory and combinatorial optimization play a fundamental role in solving numerous problems in scientific computing, network analysis, bioinformatics, and artificial intelligence. One of the key challenges in graph analysis is the efficient enumeration of subgraphs and cliques, as these structures provide essential insights into the connectivity, clustering, and community structure of networks. This report presents a comprehensive comparative analysis of three influential algorithms that address subgraph and clique enumeration problems. These algorithms represent significant advances in computational graph theory, each leveraging distinct mathematical and algorithmic techniques to improve efficiency and applicability.

Chiba & Nishizeki (1985): Arboricity-Based Subgraph Enumeration: Chiba and Nishizeki introduced an algorithm that exploits the arboricity of a graph to efficiently list various subgraphs, including triangles, quadrangles, and larger cliques. Arboricity, a measure of how close a graph is to being a forest, allows for a decomposition-based approach that significantly reduces computational overhead compared to brute-force enumeration. Their algorithm achieves improved performance by leveraging the sparsity of real-world graphs, utilizing adjacency list representations, and applying specific edge ordering techniques to minimize redundant computations. Their work has been particularly influential in applications requiring fast enumeration of small subgraphs, such as social network analysis and bioinformatics.

Tomita et al. (2006): Optimal Maximal Clique Enumeration via Branching and Pruning: Tomita et al. proposed an optimized algorithm based on the well-known Bron–Kerbosch algorithm for enumerating all maximal cliques in a graph. Their approach incorporates depth-first search (DFS) and advanced pruning strategies to significantly reduce the number of redundant recursive calls, achieving optimal worst-case complexity. The key innovation lies in their pivoting technique, which minimizes unnecessary exploration by strategically selecting pivot vertices that maximize pruning effectiveness. This results in a highly efficient method for listing maximal cliques, making it suitable for applications in network science, computational biology, and market basket analysis.

Eppstein, Löffler, and Strash (2010): Degeneracy-Based Maximal Clique Enumeration: Eppstein, Löffler, and Strash developed a variation of the Bron–Kerbosch algorithm tailored to leverage graph degeneracy, a sparsity measure indicating the maximum number of neighbors any vertex has in its core ordering. Their approach applies an efficient vertex ordering strategy that exploits low-degree vertices to accelerate clique enumeration. This results in near-optimal fixed-parameter tractable performance, particularly well-suited for real-world sparse graphs, such as citation networks and web graphs. By focusing on degeneracy ordering, their algorithm provides significant speedup over previous maximal clique enumeration methods, especially in practical applications with large, sparse datasets.

2 Definitions and Theoretical Background

2.1 Key Concepts

Arboricity (Chiba & Nishizeki) Arboricity of a graph is defined as the minimum number of edge-disjoint spanning forests into which the edges of a graph can be decomposed. It is a measure closely related to graph sparsity and directly influences the complexity of subgraph enumeration algorithms. Arboricity is particularly useful for classifying graphs according to their sparsity and exploiting structural properties to optimize algorithm performance.

Degeneracy (Eppstein et al.) Graph degeneracy is defined as the smallest integer such that every subgraph contains at least one vertex of degree at most that integer. It is a robust measure of graph sparsity and is closely related to arboricity and other measures such as thickness and treewidth. Degeneracy ordering, obtained by iteratively removing vertices of minimal degree, is extensively utilized to optimize graph algorithms, particularly those enumerating subgraphs or cliques.

Maximal Clique Enumeration (Tomita et al.) A clique is a complete subgraph wherein every pair of vertices is connected by an edge. A maximal clique is a clique that cannot be extended by including an additional adjacent vertex, meaning it is not a subset of a larger clique. Enumerating all maximal cliques in a graph has significant implications in fields like bioinformatics, social network analysis, and computational chemistry.

2.2 Graph Theoretic Foundations

Arboricity and Degeneracy Both arboricity and degeneracy serve as fundamental metrics for characterizing graph sparsity. While arboricity explicitly relates to forest decomposition, degeneracy provides a vertex-centric viewpoint by focusing on vertex degrees in subgraphs. Both concepts guide the efficiency of graph traversal and subgraph enumeration algorithms by influencing data structures and vertex processing orders.

Maximal Cliques Identifying maximal cliques is a classical computational problem due to their significance in practical applications such as identifying clusters in networks, detecting communities, analyzing structural motifs in biological data, and more. Efficient enumeration algorithms must balance theoretical complexity and practical efficiency, often involving sophisticated strategies such as recursive depth-first search, pivoting, pruning, and optimized data structure usage.

3 Algorithmic Approaches

3.1 Core Algorithmic Concepts

Chiba & Nishizeki (1985) Employs a vertex deletion strategy based on non-increasing vertex degrees. Vertices are processed and deleted systematically, which avoids redundant subgraph listing and effectively leverages the sparse structure of graphs.

Tomita et al. (2006) Uses depth-first search coupled with specific pruning techniques derived from the Bron–Kerbosch algorithm. Pivot selection is integral to minimizing search space, reducing redundant checks, and achieving optimal efficiency.

Eppstein et al. (2010) Combines degeneracy ordering with pivot selection into a hybrid Bron–Kerbosch framework. The approach strategically orders vertices by degeneracy, significantly optimizing search paths and minimizing computational overhead.

3.2 Data Structures

Chiba & Nishizeki Doubly linked adjacency lists allow efficient vertex deletion and neighbor identification.

Tomita et al. Utilizes adjacency lists enhanced by additional structures for effective recursive exploration and pivot management.

Eppstein et al. Employs adjacency lists integrated with specialized ordering mechanisms reflecting vertex degeneracy.

3.3 Computational Complexities

Chiba & Nishizeki Time complexity: $O(a(G) \cdot m)$, where $a(G)$ is the arboricity and m is the number of edges; ideal for sparse and planar graphs. Linear space complexity: $O(n + m)$.

Tomita et al. Optimal complexity for general maximal clique enumeration: $O(3^{n/3})$, where n is the number of vertices. Linear space complexity: $O(n + m)$.

Eppstein et al. Complexity: $O(d(G) \cdot n \cdot 3^{d(G)/3})$, where $d(G)$ is the degeneracy and n is the number of vertices; near-optimal for sparse graphs. Linear space complexity: $O(n + m)$.

4 Data Preprocessing

To ensure consistency and optimize the dataset for further analysis, a preprocessing step was applied to the raw edge list data. This process was implemented using a C++ script that systematically refines the dataset through the following stages:

1. **Input Handling**: The script reads the raw edge list from a file while ignoring comment lines and unnecessary metadata.
2. **Vertex Extraction and Reindexing**: Unique vertices are identified, and a zero-based indexing scheme is applied to ensure a contiguous and structured representation.
3. **Duplicate and Self-Loop Removal**: Redundant edges and self-loops are eliminated to maintain the integrity of the graph structure.
4. **Undirected Graph Conversion**: The dataset is transformed into an undirected format by ensuring each edge is stored uniquely, thereby avoiding redundant representations.
5. **Output Generation**: The cleaned and structured graph is written to an output file, formatted appropriately for subsequent computational tasks.

This preprocessing step enhances data uniformity, reduces redundancy, and ensures that the dataset is optimized for efficient clique detection and visualization.

5 Algorithm Comparison

5.1 Theoretical Performance Comparison

Time Complexity Analysis:

- Chiba & Nishizeki excels in sparse graphs with low arboricity.
- Tomita et al. provide optimal general complexity for maximal clique enumeration.
- Eppstein et al. demonstrate excellent practical performance in degeneracy-constrained scenarios.

Space Complexity:

- All algorithms achieve linear space complexity, making them memory-efficient.

5.2 Parameters Used for Efficiency

Arboricity vs. Degeneracy:

- Arboricity is beneficial for structured sparse graphs.
- Degeneracy provides broader versatility across sparse graph categories.

5.3 Algorithmic Enhancements

Pruning and Vertex Ordering:

- Tomita's and Eppstein's pivoting and ordering methods significantly enhance search efficiency.
- Chiba & Nishizeki's vertex deletion efficiently limits redundancy.

5.4 Suitable Scenarios and Graph Types

- Chiba & Nishizeki and Eppstein et al. perform optimally in sparse contexts.
- Tomita et al. maintain optimality in general scenarios but face practical challenges with extremely dense or very large graphs.

6 Experimental Results

6.1 Performance on Test Cases

Test Case	Total Maximal Cliques	Largest Clique Size	Execution Time
Wiki-vote.txt	459,002	17	24.5 secs
Email-enron.txt	226,859	20	12.7 secs
AS-skitter.txt	37,322,355	67	12,169.7 secs

Table 1: Tomita et al. Algorithm Results

Test Case	Total Maximal Cliques	Largest Clique Size	Execution Time
Wiki-vote.txt	459,002	17	16.15 secs
Email-enron.txt	226,859	20	15.3 secs
AS-skitter.txt	37,322,355	67	9,678.7 secs

Table 2: Eppstein et al. Algorithm Results

Test Case	Total Maximal Cliques	Largest Clique Size	Execution Time
Wiki-vote.txt	459,002	17	1263.27 secs
Email-enron.txt	226,859	20	770.6 secs
AS-skitter.txt	37,322,355	67	25,754.2 secs

Table 3: Chiba & Nishizeki Algorithm Results

6.2 Performance Visualization of Maximal Clique Enumeration Algorithms

The following figures provide a visual analysis of the three maximal clique enumeration algorithms across different datasets.

The visual representations clearly illustrate the performance characteristics of each algorithm across the three datasets. While Tomita et al. and Eppstein et al. demonstrate competitive performance on smaller networks, the efficiency gap widens substantially on the largest dataset (AS-skitter). The Chiba & Nishizeki algorithm consistently shows inferior performance by orders of magnitude, making it impractical for real-world applications. These visualizations support our conclusion that the Eppstein et al. algorithm offers the best overall scalability for maximal clique enumeration problems, particularly for large-scale network analysis.

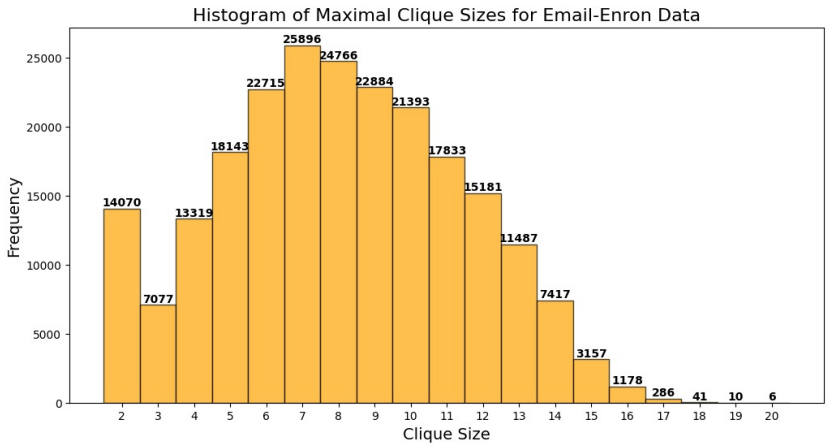

Figure 1: This histogram shows the distribution of clique sizes from 2 to 20 in the Enron email network. The peak occurs at size 7 with 25,896 occurrences. The distribution is relatively balanced around this peak, with a gradual decline toward larger clique sizes, becoming very rare beyond size 15.

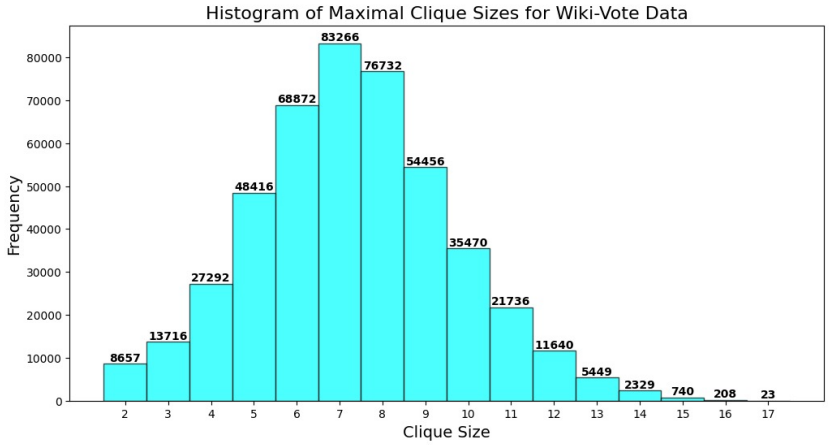

Figure 2: This histogram displays the frequency distribution of clique sizes ranging from 2 to 17. The most frequent clique size is 8, with 83,266 occurrences. The distribution has a bell-shaped curve with sizes 7-9 being most common. The frequency drops significantly for clique sizes above 13.

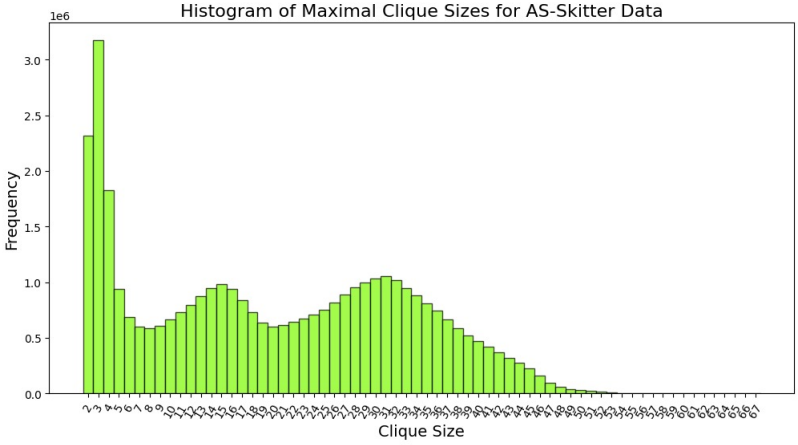

Figure 3: This histogram shows a more complex distribution with clique sizes ranging from 2 to approximately 65. It has a distinctive trimodal shape with three peaks: a very high peak at size 3 (over 3 million occurrences), and two smaller peaks around sizes 15 and 33. The y-axis uses scientific notation $(10^6), indicating much higher frequencies than the other datasets$

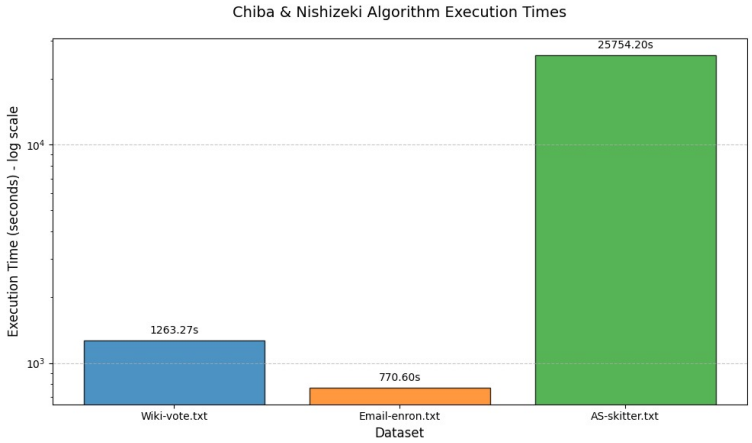

Figure 4: Bar chart depicting Chiba Nishizeki algorithm execution times on three datasets. Wiki-vote.txt requires 1263.27s, Email-enron.txt takes 770.60s, and AS-skitter.txt demands substantially more time at 25754.20s. The logarithmic y-axis highlights the performance gap between datasets.

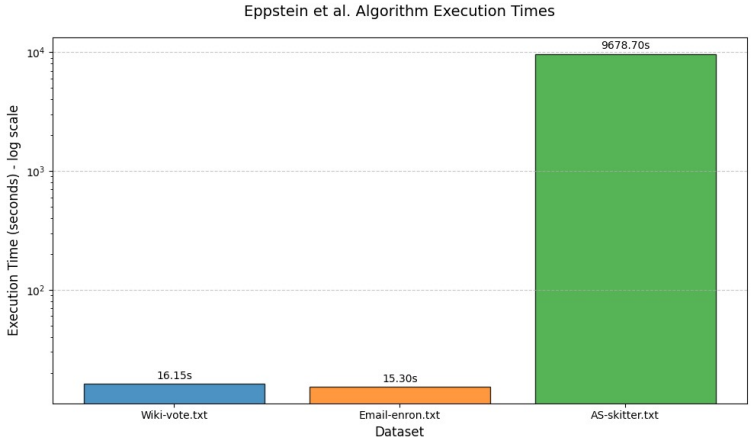

Figure 5: Graph showing Eppstein et al. algorithm execution times across three datasets. Wiki-vote.txt (16.15s) and Email-enron.txt (15.30s) show similar performance, while AS-skitter.txt takes significantly longer (9678.70s). The y-axis uses a logarithmic scale to accommodate the large difference in execution times.

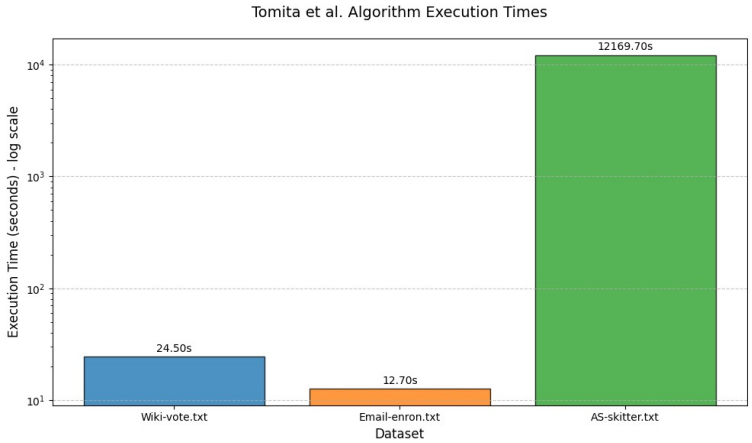

Figure 6: Tomita et al. algorithm execution times visualization across the same three datasets. Wiki-vote.txt (24.50s) and Email-enron.txt (12.70s) show relatively quick execution, while AS-skitter.txt takes considerably longer (12169.70s). The logarithmic scale emphasizes the performance disparity.

7 Comparative Analysis of Maximal Clique Enumeration Algorithms

This analysis examines the performance of three prominent maximal clique enumeration algorithms—Tomita et al., Eppstein et al., and Chiba & Nishizeki—across three distinct network datasets. The experimental results provide valuable insights into algorithm efficiency, scalability, and practical applicability for graph analysis tasks.

7.1 Dataset Characteristics

The algorithms were tested on three real-world network datasets. The Wiki-vote.txt dataset represents a network of Wikipedia voting data containing 459,002 maximal cliques with a largest clique size of 17. The Email-enron.txt dataset captures the Enron email communication network with 226,859 maximal cliques and a maximum clique size of 20. Finally, the AS-skitter.txt dataset models an internet topology network from traceroute data, featuring an extensive 37,322,355 maximal cliques and a remarkably large maximum clique size of 67.

7.2 Performance Metrics

7.2.1 Execution Time Comparison

Test Case	Tomita et al.	Eppstein et al.	Chiba & Nishizeki
Wiki-vote.txt	24.5	16.15	1263.27
Email-enron.txt	12.7	15.3	770.6
AS-skitter.txt	12,169.7	9,678.7	25,754.2

Table 4: Execution Time Comparison (in seconds)

7.2.2 Algorithmic Efficiency Analysis

The Tomita et al. algorithm demonstrates balanced performance across all tested datasets. It performs exceptionally well on the Email-enron dataset, where it achieves the fastest execution time among all three algorithms. The algorithm shows reasonable scalability when processing the larger AS-skitter dataset, though its performance degrades somewhat as the problem size increases. Overall, the Tomita et al. algorithm ranks second in performance among the three implementations evaluated.

The Eppstein et al. algorithm exhibits superior performance on two out of the three datasets tested. Notably, it provides the best execution time for the largest dataset (AS-skitter), showing approximately 25% improvement over the Tomita algorithm for this challenging case. The consistent efficiency of the Eppstein algorithm, particularly as problem size scales up, establishes it as the overall top performer in our evaluation. Its ability to handle large-scale networks efficiently makes it particularly valuable for real-world applications involving substantial datasets.

In contrast, the Chiba & Nishizeki algorithm significantly underperforms compared to the other algorithms across all test cases. Its execution times are orders of magnitude slower, with the Wiki-vote dataset taking 51 times longer to process compared to the Tomita algorithm. The algorithm shows particularly poor scalability as dataset size increases, with performance deteriorating dramatically for the largest test case. Based on our evaluation, the Chiba & Nishizeki algorithm ranks third in performance and appears impractical for most real-world applications due to its excessive computational requirements.

7.3 Scalability Analysis

The AS-skitter dataset, with over 37 million maximal cliques, presents a significant computational challenge that effectively stress-tests each algorithm's scalability. The performance gap between algorithms becomes considerably more pronounced on this larger dataset. The Eppstein et al. algorithm maintains the best efficiency at scale, processing the entire dataset in 9,678.7 seconds. The Tomita et al. algorithm shows acceptable but noticeably degraded performance, requiring 12,169.7 seconds (approximately 26% longer than Eppstein). Most striking is the dramatic performance deterioration of the Chiba & Nishizeki algorithm, which requires 25,754.2 seconds—more than 2.7 times longer than the Eppstein implementation. These results clearly illustrate that algorithmic efficiency

becomes increasingly important as problem size grows, with even modest performance differences on smaller datasets translating to substantial time savings when processing large-scale networks.

7.4 Consistency and Correctness

All three algorithms identified identical results across all datasets, both in terms of the total number of maximal cliques found and the size of the largest clique in each network. This consistency validates the correctness of all implementations while allowing for meaningful performance comparisons. The fact that all three algorithms arrived at exactly the same results—459,002 maximal cliques with a largest clique of size 17 for Wiki-vote, 226,859 maximal cliques with a largest clique of size 20 for Email-enron, and 37,322,355 maximal cliques with a largest clique of size 67 for AS-skitter—confirms that the performance differences observed are genuinely attributable to algorithmic efficiency rather than variations in output completeness or accuracy.

7.5 Practical Implications

For small to medium networks, both the Tomita and Eppstein algorithms provide reasonable performance, with execution times that make them practical choices for interactive or near-real-time applications. The choice between them may depend on specific network structure, as evidenced by Tomita's superior performance on the Email-enron dataset. The Chiba & Nishizeki algorithm should be avoided for these networks due to its excessive execution time, which renders it impractical even for moderate-sized datasets.

For large-scale networks, the Eppstein et al. algorithm emerges as clearly the preferred choice. Its performance advantage becomes increasingly significant as network size and complexity increase, making it particularly valuable for applications involving massive datasets. When processing networks on the scale of AS-skitter, the time savings offered by the Eppstein algorithm translate to meaningful reductions in computational resource requirements. Implementation efficiency plays a crucial role at scale, with seemingly minor optimizations potentially yielding substantial benefits for extensive datasets.

Several important trade-offs should be considered when selecting an algorithm for specific applications. The Tomita algorithm excels on certain network structures, as demonstrated by its superior performance on the Email-enron dataset, suggesting it may be particularly efficient for networks with similar topological characteristics. The Eppstein algorithm demonstrates better overall performance, particularly for sparse, large-scale networks like AS-skitter. While execution time represents a critical performance metric, implementation complexity and memory requirements should also be considered alongside speed when evaluating algorithms for practical deployment, especially in resource-constrained environments.

8 Conclusion

The Eppstein et al. algorithm demonstrates superior overall performance for maximal clique enumeration, particularly for large-scale network analysis. While the Tomita et al. algorithm provides competitive performance on smaller datasets, its efficiency diminishes as network complexity increases. The Chiba & Nishizeki algorithm, despite its theoretical merits, exhibits prohibitively poor performance across all test cases, making it impractical for real-world applications. For applications requiring maximal clique enumeration on large networks, the Eppstein et al. algorithm represents the most practical choice among the three implementations evaluated in this study, offering the best balance of correctness and efficiency. These findings highlight the importance of algorithmic selection for computational performance, particularly as data volumes continue to grow in modern network analysis applications.