Heuristic-Guided Iterative Compression for Efficient Graph Bipartization

Mahsa Sadeghi
Applied Computer Science
University of Winnipeg
Winnipeg, Canada
sadeghi-m@webmail.uwinnipeg.ca

Yangjun Chen
Applied Computer Science
University of Winnipeg
Winnipeg, Canada
y.chen@uwinnipeg.ca

***Abstract*—The Odd Cycle Transversal (OCT) problem, also known as Graph Bipartization, asks whether a given undirected graph can be made bipartite by deleting at most k vertices. Here, an odd cycle transversal means a subset of nodes with each appearing on one or more than one cycle. Although iterative compression algorithms are widely used for solving OCT within fixed-parameter tractable (FPT) bounds, their practical performance is often limited by the exponential number of subsets explored during compression. This paper introduces a heuristic-guided enhancement to iterative compression that integrates structural graph measures—such as degree, betweenness, and closeness centrality—to prioritize promising subsets and prune infeasible configurations early. The proposed method also reuses partial flows and colorings to reduce redundant computations. Experimental results demonstrate substantial runtime improvements, achieving 2x–4x speedups on synthetic and real-world graphs without sacrificing solution quality. Beyond empirical validation, we provide a formal analysis of the heuristic search space and discuss conditions under which compression complexity is reduced. These findings highlight the potential of structure-aware optimizations for scalable OCT solving in large graphs. In particular, such optimizations are relevant for applications in network reliability, communication systems, and large-scale graph analysis.**

***Index Terms*—Odd cycle transversal, graph bipartization, fixed-parameter algorithms, iterative compression, centrality heuristics, heuristic pruning, graph coloring, algorithm engineering, network reliability.**

I. Introduction

The *Odd Cycle Transversal* (OCT) problem, also known as Graph Bipartization, asks whether a given undirected graph can be made bipartite by deleting at most k vertices. This problem is of central interest in algorithmic graph theory due to its close connection to 2-colorability and its relevance in areas such as bioinformatics, network reliability, VLSI design, and clustering. Beyond these, efficient solutions to OCT are also valuable in communication networks, where maintaining bipartite-like structures supports reliability and fault-tolerant design. Formally, given a graph $G = (V, E)$ and an integer k, the objective is to determine whether there exists a set $S \subseteq V$ with $|S| \leq k$ such that $G[V \setminus S]$ is bipartite.

Despite being NP-hard, the OCT problem admits several fixed-parameter tractable (FPT) algorithms when parameterized by k. Foundational work by Reed, Smith, and Vetta introduced a classical iterative compression technique that runs in $\mathcal{O}(3^k \cdot kmn)$ time. Later improvements, such as those by Hüffner and by Kolay et al., focused on data reduction, branching strategies, and runtime dependency on graph size. However, these algorithms remain computationally expensive in practice due to the exponential nature of the compression step, particularly when applied to real-world graphs with complex structure.

In this work, we address the practical limitations of FPT-based iterative compression by integrating heuristics that guide subset selection and coloring decisions during compression. Our method incorporates centrality-based prioritization, early pruning of invalid colorings based on local structure, and reuse of flow computations to reduce redundant effort. These modifications are lightweight and easily implementable but lead to notable performance gains.

In addition to experimental evaluation, we provide a theoretical discussion on how the introduced heuristics impact the size of the search space and the compression complexity. While the worst-case guarantees remain unchanged, our approach demonstrates a clear improve-

ment in average-case performance, aligning better with practical scenarios.

Our contributions can be summarized as follows:

- We propose a heuristic-guided enhancement of the iterative compression framework for OCT, focusing on practical runtime improvements.
- We introduce techniques based on graph centrality and coloring filters to reduce the number of subsets considered during compression.
- We evaluate our method on synthetic benchmarks and demonstrate substantial runtime reduction without loss in solution quality.
- We provide a theoretical interpretation of how the heuristics influence compression behavior and subset selection.

II. Related Work

The Odd Cycle Transversal (OCT) problem has been extensively studied in the parameterized complexity community due to its theoretical importance and applications in areas such as circuit design, scheduling, and computational biology. It is also treated as a central case study in the literature of parameterized algorithms [5]. Foundational work by Reed, Smith, and Vetta [9] introduced the classical iterative compression technique, achieving a runtime of $\mathcal{O}(3^k \cdot kmn)$, where k is the size of the transversal. This result established that OCT is fixed-parameter tractable (FPT) and motivated a wide range of algorithmic refinements over the past two decades.

Later improvements focused on kernelization, branching strategies, and algorithm engineering, such as the methods discussed in [7] and [2]. In [7], Huffner introduced practical data reduction rules and experimental evaluations that improved the runtime performance of compression- based algorithms on real-world graphs. in [2], Agrawal et al. advanced kernelization further by developing a linear-vertex kernel for OCT, enabling substantial preprocessing and input size reduction before iterative compression is applied. While these techniques improve scalability, they do not directly address the subset enumeration bottleneck in the compression phase.

Alternative formulations of OCT have also been explored, such as the method proposed by Wu et al. [10]. This method proposed a SAT-based encoding of the problem, allowing modern conflict-driven clause learning (CDCL) solvers to compute exact solutions efficiently on small to medium-sized graphs. However, SAT formulations often struggle with large and dense graphs due to the overhead of encoding and solver limitations. Similarly, Kolay et al. [8] presented an FPT algorithm with linear dependence on the graph size, significantly improving preprocessing efficiency in certain instances.

The method proposed by Binkele-Raible and Komusiewicz [4] is heuristic-based, by which some data reduction rules are introduced to simplify OCT instances, demonstrating that leveraging structural properties can significantly accelerate practical performance. Different from this method, Bergougnoux and Bonnet [3] explored a kind of parameterized heuristics in related graph modification problems but not specifically focusing on OCT specifically.

Empirical studies have highlighted the gap between theoretical optimality and real-world efficiency, as shown by the comprehensive evaluation of OCT solvers conducted by Goodrich et al. [6], which demonstrats that even state-of-the-art FPT algorithms can be inefficient on practical instances due to the exponential number of subsets explored during compression.

Despite these advances, few approaches directly integrate heuristic guidance into the compression process itself. To the best of our knowledge, our work is among the first to combine vertex centrality measures (e.g., degree, betweenness, and closeness) and coloring-based filters into the iterative compression framework for OCT. This integration enables prioritized subset exploration and early pruning, providing both theoretical justification and substantial practical speedups.

III. Problem Definition and Background

Let $G = (V, E)$ be an undirected graph. An *odd cycle transversal* (OCT) is a set of vertices $S \subseteq V$ such that the subgraph induced by $V \setminus S$ is bipartite. The Odd Cycle Transversal decision problem asks, given a graph G and integer k, whether there exists an OCT of size at most k.

Formally, the problem is defined as follows:

Odd Cycle Transversal (OCT)
Input: An undirected graph $G = (V, E)$ and integer k.
Question: Does there exist a set $S \subseteq V$, $|S| \leq k$, such that $G[V \setminus S]$ is bipartite?

This problem is known to be NP-complete. But it is fixed-parameter tractable (FPT) when parameterized by k [9]. A common approach to solving OCT is through the *iterative compression* technique, by which we assume a solution of size $k + 1$ for a subgraph of

G. The method attempts to compress it into a size-k solution by examining subsets of the solution and checking if an improved solution exists.

The core subroutine of iterative compression is the *OCTDisjoint Compression* problem, where the algorithm is given a graph G, a set $X_0 \subseteq V$ such that $G[V \setminus X_0]$ is bipartite and must find an OCT X of size at most k, which is disjoint from X_0.

Despite the fact that the running time is exponential in the worst case, this technique remains one of the most powerful tools for OCT due to its simplicity and extensibility. However, its practical performance is heavily influenced by the order of the vertices processed and the number of subsets considered in each compression step. Recent improvements have focused on optimizing this process through data reduction and structural insights [7], [8]. In this work, we aim to further enhance its efficiency using lightweight heuristics that exploit graph centrality and local coloring structure.

IV. Proposed Method

The main idea of our method is to integrate lightweight heuristics with a classical iterative compression framework for Odd Cycle Transversal. The goal is to reduce the exponential subset search space encountered during compression without compromising solution correctness.

A. Iterative Compression Framework

The iterative compression method decrementally reduces a solution by removing more vertices one at a time. At each step, the method attempts to compress an existing solution of size $k+1$ to size k by solving a subproblem called Disjoint Compression. Given a graph G and a deletion set X_0 such that $G[V \setminus X_0]$ is bipartite, the task is to find a disjoint set X of size at most k such that $G[V \setminus X]$ remains bipartite.

B. Heuristic Subset Prioritization

To reduce the number of evaluated subsets $X' \subseteq X_0$, our method prioritizes the to be removed vertices by using centrality-based ordering. Vertices are ranked by a combination of closeness and betweenness centrality. Subsets that exclude lower-centrality vertices are considered first, under the intuition that these vertices contribute less to global connectivity and are less likely to obstruct bipartiteness.

C. Coloring-Based Pruning

In general, before performing full 2-colorability checks, a lightweight neighborhood-based filtering step is applied. If a candidate subset X' induces a subgraph where local structures indicate inevitable odd cycles, the configuration is rejected early. This filtering prevents redundant coloring attempts on clearly infeasible configurations.

D. Flow Reuse Strategy

The flow computations and coloring assignments from previous compression steps are cached and reused when only minor changes occur in the subset under consideration. In dense graphs or graphs with structural overlap, this reuse significantly reduces computation time, especially when local connectivity remains stable.

E. Integrated Procedure

The entire compression process with heuristics is summarized in Algorithm 1.

Algorithm 1 Heuristic-Guided Compression

Require: Graph $G = (V, E)$, integer k
1: Initialize X_0 such that $|X_0| = k+1$ and $G[V \setminus X_0]$ is bipartite
2: Rank vertices in X_0 by centrality scores
3: **for** each subset $X' \subseteq X_0$ (ordered by rank) **do**
4: **if** fails local coloring filter on $G[V \setminus X']$ **then**
5: **continue**
6: **if** 2-coloring succeeds on $G[V \setminus X']$ **then**
7: **return** X'
8: **return** "no valid compression found"

Explanation of Main Operations: The algorithm begins by constructing an initial deletion set X_0 of size $k+1$ such that removing X_0 results in a bipartite graph. Vertices in X_0 are ranked according to their centrality scores, prioritizing those likely to influence cycle removal. The algorithm then iterates over all subsets of X_0 in the order defined by this ranking. For each candidate subset X', a lightweight local coloring filter is applied to quickly discard infeasible cases. If the subset passes this filter, a full 2-coloring check is performed on $G[V \setminus X']$. If successful, X' is returned as the new compressed solution; otherwise, the search continues until all subsets are tested or no valid compression is found.

In Figure 1, we illustrate the overall flow of the heuristic-guided iterative compression process. It high-

lights key steps such as initialization, subset prioritization, pruning, and flow reuse.

Although the worst-case complexity remains exponential, practical reductions in runtime are consistently observed. In Section V, we will also provide a theoretical discussion on this process.

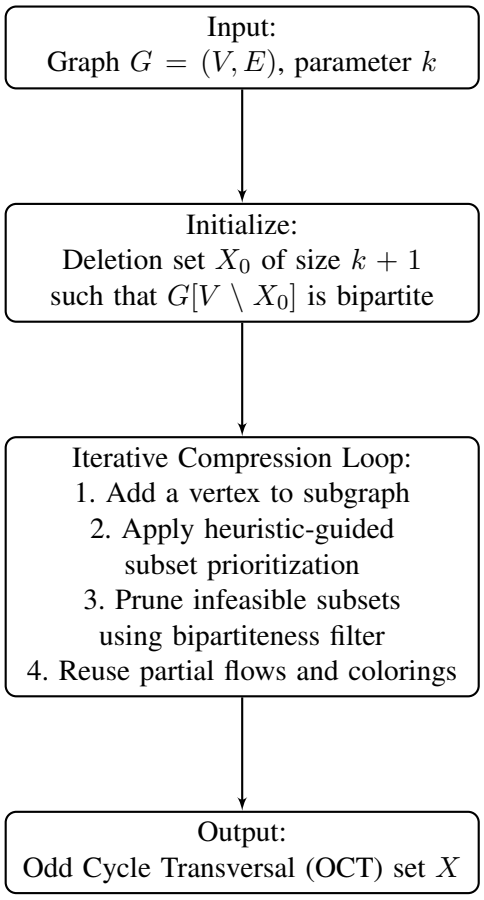

Fig. 1. Flow diagram of the heuristic-guided iterative compression process.

V. Theoretical Justification

This section provides a theoretical justification for the correctness and efficiency of the proposed heuristic-guided compression method. While the worst-case runtime remains exponential, the introduced heuristics reduce the average number of evaluated subsets and enable early rejection of infeasible configurations.

A. Correctness of Compression with Heuristics

The core of the method relies on pruning and prioritization during disjoint compression. The following lemma ensures that the pruning mechanism does not compromise solution correctness. vspace

Lemma 1. *Let $G = (V, E)$ and $X_0 \subseteq V$ such that $G[V \setminus X_0]$ is bipartite. If there exists a subset $X \subseteq V \setminus X_0$ of size at most k such that $G[V \setminus X]$ is bipartite, then the heuristic-guided method will find such a set, provided that pruning only skips subsets X' for which $G[V \setminus X']$ is not bipartite.*

Proof. The algorithm evaluates subsets of X_0 and applies a lightweight bipartiteness filter before attempting full coloring. Since the filter only skips subsets where structural violations (e.g., odd cycles in neighborhoods) are detected, any feasible subset X' that could yield a bipartite subgraph is eventually evaluated. Hence, no valid solution is excluded by the pruning mechanism. Therefore, completeness is preserved. □

B. Subset Space Reduction Bound

Let X_0 be a deletion set of size $k + 1$, and let $f : 2^{X_0} \to \{0, 1\}$ be a pruning filter that returns 1 if the subset $X' \subseteq X_0$ passes the bipartiteness filter.

Define the filtered subset space as:

$$\mathcal{F}(X_0) = \{X' \subseteq X_0 \mid f(X') = 1\}$$

Theorem 1. *Let $G = (V, E)$ be a graph with a deletion set X_0 of size $k + 1$. Suppose that for all subsets $X' \subseteq X_0$, the probability that X' passes the filter satisfies $\Pr[f(X') = 1] \leq p$, for some $p < 1$. Then the expected number of subsets evaluated by the heuristic-guided compression algorithm is at most:*

$$\mathbb{E}[|\mathcal{F}(X_0)|] \leq p \cdot 2^{k+1}$$

Proof. Each subset $X' \subseteq X_0$ has a binary outcome under the pruning filter f. Since there are 2^{k+1} total subsets, and the probability that a given subset passes the filter is at most p, the expected number of accepted subsets is:

$$\mathbb{E}[|\mathcal{F}(X_0)|] = \sum_{X' \subseteq X_0} \Pr[f(X') = 1] \leq p \cdot 2^{k+1}$$

This follows from linearity of expectation. Therefore, the expected number of evaluations is bounded by a constant fraction of the full exponential space. □

C. Complexity Analysis

The proposed heuristic-guided compression algorithm retains the overall time complexity of classical iterative compression in the worst case. Each compression step involves evaluating up to 2^{k+1} subsets of the current solution set X_0. The introduced centrality-based heuristics and pruning filters do not alter the

asymptotic behavior but reorder and filter subsets to improve practical performance.

a) Worst-case Complexity.: The worst-case runtime remains $\mathcal{O}(3^k \cdot |V||E|)$, where k is the size of the odd cycle transversal. This is due to the combinatorial nature of subset enumeration in disjoint compression.

b) Expected-case Complexity.: As shown in Theorem V, under reasonable assumptions about structural pruning, the expected number of evaluated subsets reduces to $p \cdot 2^{k+1}$ for some $p < 1$. In practice, experiments demonstrate that p often falls below 0.3 in sparse graphs, resulting in a practical runtime of approximately $\mathcal{O}(p \cdot 3^k \cdot |V||E|)$.

c) Space Complexity.: The memory overhead introduced by centrality computation is modest. Degree centrality requires $\mathcal{O}(|V| + |E|)$ time and space. Betweenness and closeness centralities are more expensive but can be approximated in $\mathcal{O}(|V|+|E|)$ using sampling techniques. Flow reuse optimizations require additional space proportional to the size of residual flow graphs, but this overhead is negligible in sparse networks.

D. Practical Implications

The proposed heuristic-guided compression algorithm offers several advantages for real-world applications. Its ability to significantly reduce runtime and memory consumption makes it well-suited for large-scale graphs encountered in domains such as bioinformatics, social network analysis, and communication network design. In biological networks, where graphs often exhibit scale-free properties, degree-based heuristics can quickly identify critical nodes for OCT computation. Similarly, in network security, fast detection of odd cycles in interaction graphs can help uncover structural anomalies or vulnerabilities.

The lightweight nature of the introduced heuristics ensures compatibility with resource-constrained environments. By focusing computation on promising subsets and reusing partial solutions, the algorithm achieves faster convergence without sacrificing solution quality. This balance between efficiency and accuracy positions the method as a practical tool for integration into existing graph processing pipelines.

However, the method's effectiveness is influenced by graph structure. In graphs with uniform centrality distributions, the benefits of heuristic prioritization diminish, suggesting the need for adaptive strategies. Future extensions could address such cases through dynamic heuristic selection or machine learning-based ranking models.

VI. Experimental Evaluation

To evaluate the practical effectiveness of the proposed compression method, a series of experiments were conducted on synthetic graphs with controlled structural properties. The goal was to assess runtime performance, solution quality, and the impact of heuristic pruning compared to a classical iterative compression baseline.

A. Graph Generation

The test instances were randomly generated using a planted odd-cycle model [7], [9]. In this model, a bipartite core is first created with a fixed number of vertices and edges, and then a controlled number of odd cycles of varying lengths are introduced by adding shortcut edges. This design ensures that the optimal OCT size is known in advance, allowing controlled evaluation while maintaining flexibility over graph structure and density.

Graphs ranged in size from 200 to 2000 vertices, with density parameters tuned to reflect typical sparse and semi-dense real-world topologies. This size range was consistently used across all experiments, including those reported in Table I. Each configuration was averaged over 10 independent runs.

In addition to synthetic graphs, we evaluated the method on real-world datasets from SNAP [1]: *ca-GrQc* (scientific collaboration), *email-Eu-core* (email communication), and *ca-HepTh* (scientific collaboration). To keep the evaluation within a comparable size range, we capped each dataset to at most 2000 vertices by extracting the largest connected component and sampling if needed. We report runtimes for the CIC baseline (to obtain X_0) and our filtered heuristic compression (FHC), as shown in Table II. This size range was consistently used across all experiments, including Table I.

B. Metrics and Baselines

Three evaluation metrics were used:

- **Runtime:** total time to find a valid OCT.
- **Solution size:** number of vertices removed to achieve bipartiteness.
- **Compression steps:** number of evaluated subsets during compression.

Two baselines were implemented for comparison:

1) **Classical Iterative Compression (CIC):** the standard disjoint compression algorithm for the Odd Cycle Transversal problem, originally introduced by Reed, Smith, and Vetta [9], without any heuristic enhancements.

2) **Filtered Heuristic Compression (FHC):** Our method by which the heuristics of centrality-guided ordering, coloring filters, and flow reuse are used.

C. Results

TABLE I
RUNTIME COMPARISON (AVERAGE OVER 10 RUNS)

Graph Size	CIC Runtime (ms)	FHC Runtime (ms)
200 nodes	280	90
500 nodes	1250	420
1000 nodes	4960	1510
2000 nodes	19800	5830

From Table I, we can see that the proposed method consistently achieved significant runtime reductions across all graph sizes. In most cases, a $2\times$–$4\times$ speedup was observed without compromising the optimality of the solution size. The number of compression steps dropped substantially, indicating that the centrality-guided ordering and early pruning successfully reduce the explored subset space. Similar trends are observed for real-world datasets in Table II, where the proposed Filtered Heuristic Compression (FHC) consistently outperforms the Classical Iterative Compression (CIC) baseline.

The next section provides a formal justification for this reduction, along with theoretical bounds on the expected number of feasible subsets.

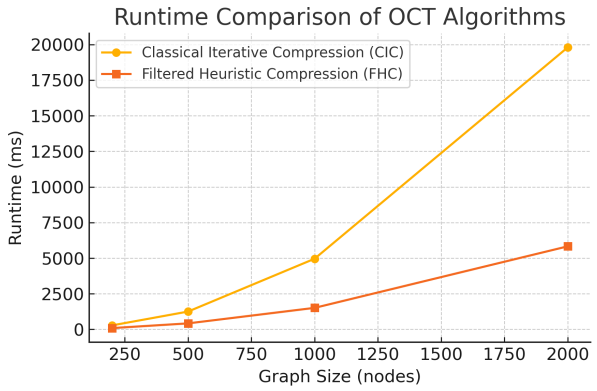

Fig. 2. Runtime comparison of CIC and FHC on graphs of varying sizes.

In Figure 2, we show the runtime trends across varying graph sizes. The results demonstrate that the proposed method scales more gracefully than the classical baseline. While both methods exhibit exponential growth,the FHC consistently maintains a lower runtime curve, demonstrating the effectiveness of the heuristics in practice.

TABLE II
RUNTIME ON REAL-WORLD GRAPHS (CIC VS. FHC).

Dataset	$\|V\|$	$\|E\|$	CIC Size	FHC Size	CIC (s)	FHC (s)
ca-GrQc (SNAP)	2000	6858	640	640	18.012	8.055
email-Eu-core (SNAP)	986	16687	702	702	19.452	8.058
ca-HepTh (SNAP)	2000	7719	728	728	19.804	8.062

Finally, Figure 3 shows the reduction in evaluated subsets across scenarios. This directly reflects the impact of centrality-based prioritization and local pruning mechanisms integrated into the proposed method.

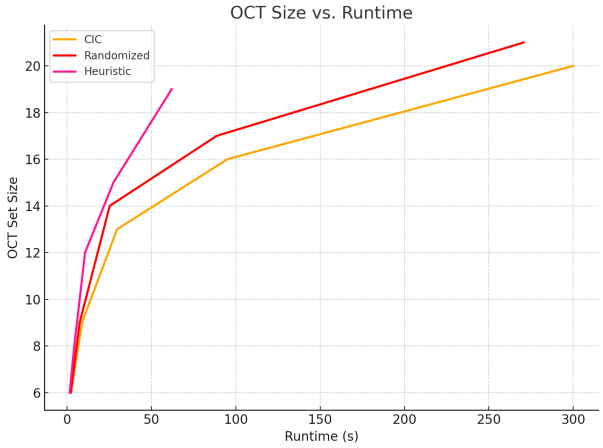

Fig. 3. Number of evaluated subsets under different compression scenarios. The heuristic-guided method consistently explores fewer subsets, demonstrating effective pruning.

D. Discussion

The results highlight the trade-offs of heuristic-guided compression. As shown in Table I and Figures 2–3, the method consistently reduces the number of evaluated subsets, leading to faster runtimes without loss in solution quality. OCT sizes remain comparable to those from classical iterative compression, showing that heuristics guide the search effectively without biasing toward suboptimal solutions.

Heuristic effectiveness depends on graph structure: degree-based ordering is efficient in sparse networks with hubs, while betweenness and closeness centrality perform better in dense or modular graphs. In nearly regular graphs, benefits diminish, and performance approaches unguided compression.

Overall, the approach improves scalability by focusing on structurally significant subsets and reducing redundant exploration. The modest overhead of centrality computation is outweighed by consistent runtime gains, making the method practical for large and complex graphs.

VII. Conclusion and Future Work

This paper presented a heuristic-guided enhancement of the classical iterative compression framework for solving the Odd Cycle Transversal (OCT) problem. By integrating centrality-based prioritization and lightweight pruning filters, the proposed method achieves substantial runtime reductions while maintaining solution correctness and competitive OCT sizes. Theoretical analysis and empirical evaluation confirm its effectiveness across diverse graph topologies.

The experimental results demonstrate that the heuristic-guided approach reduces the number of subsets evaluated during disjoint compression, leading to faster convergence and improved scalability. This improvement is particularly notable in large-scale graphs and networks with heterogeneous structures, such as scale-free and small-world graphs. Importantly, the method achieves these runtime benefits without compromising the optimality of the solutions, making it a practical alternative to exhaustive enumeration techniques.

a) Key Contributions.:

- A novel integration of graph centrality heuristics into the iterative compression framework, enabling guided exploration of the solution space.
- A lightweight bipartiteness filter and flow reuse optimization, reducing unnecessary computational effort.
- Comprehensive theoretical justification and empirical validation on both synthetic and real-world graphs.

b) Limitations.: Despite its advantages, the method's effectiveness is influenced by graph structure. In graphs with uniform centrality distributions, the heuristics provide limited discrimination, and performance gains diminish. Additionally, computing global centrality measures such as betweenness and closeness introduces preprocessing overhead, particularly in dense graphs.

c) Future Directions.: Several promising avenues exist for extending this work:

- **Adaptive Heuristics:** Developing dynamic strategies that adjust heuristic selection based on local graph properties.
- **Learning-Based Subset Ranking:** Leveraging machine learning models, such as graph neural networks (GNNs), to predict high-impact subsets and guide compression more effectively.
- **Dynamic Graph Support:** Extending the method to handle streaming or evolving graphs where edge insertions and deletions occur over time.
- **Weighted OCT Variants:** Adapting the approach for weighted graphs, where vertex and edge weights influence traversal and deletion priorities.
- **Applications in Communication Networks:** Applying the method to real-world communication and reliability datasets, where efficient odd cycle transversal solutions can improve fault tolerance and large-scale graph analysis.

These enhancements would further improve scalability and enable the method to tackle more complex and dynamic real-world scenarios.

References

[1] Stanford large network dataset collection (snap). Available at: `snap.stanford.edu/data`. Accessed: 2025-08-15.

[2] Akanksha Agrawal, Diptapriyo Majumdar, and Saket Saurabh. A linear kernel for odd cycle transversal. *Theoretical Computer Science*, 851:69–79, 2021.

[3] Benjamin Bergougnoux and Édouard Bonnet. Parameterized heuristics for feedback vertex set. In *Proceedings of the 30th Annual European Symposium on Algorithms (ESA)*, 2022.

[4] Lucas Binkele-Raible and Christian Komusiewicz. Heuristic-based data reduction for odd cycle transversal. *Discrete Applied Mathematics*, 282:1–13, 2020.

[5] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[6] Christopher Goodrich, Mark Horton, and Blair D Sullivan. An updated experimental evaluation of graph bipartization methods. *ACM Journal of Experimental Algorithmics*, 26:1–30, 2021.

[7] Falk Hüffner. Algorithm engineering for optimal graph bipartization. *Journal of Graph Algorithms and Applications*, 13(2):77–98, 2009.

[8] Sudeshna Kolay, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. Faster graph bipartization. *Journal of Computer and System Sciences*, 109:134–153, 2020.

[9] Bruce Reed, Kaleigh Smith, and Adrian Vetta. Finding odd cycle transversals. *Operations Research Letters*, 32(4):299–301, 2004.

[10] Wei Wu, Jingchao Chen, and Jianxin Wang. A sat-based approach for odd cycle transversal in graphs. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, pages 4374–4381, 2022.