Quest: Journal of Geometry, Mathematical and Quantum Physics
e-ISSN: 3064-7282 Vol.2 Issue 5|2025

Optimizing Safe Chemical Storage Using Graph Coloring Algorithms and Analysis

S. S. Turdiev, D. E. Ibrohimova
Assistant of the Department of Higher Mathematics, Tashkent State Transport University, Uzbekistan

Abstract: This paper investigates the problem of safe chemical storage by leveraging vertex graph coloring. Chemical substances are modeled as vertices, with potential reactions represented as edges in an undirected graph. The objective is to minimize the number of storage groups (colors) to prevent interactions between reactive substances. We analyze two algorithms: a greedy algorithm and the DSATUR heuristic, and propose a modified approach incorporating local optimization. Numerical experiments are conducted on graphs of varying sizes and densities, with results visualized through graphs and tables. The modified algorithm reduces the number of colors by 10–20% compared to the greedy approach, maintaining acceptable computational complexity. The findings highlight the potential for practical implementation in chemical laboratories.

Keywords: Graph coloring, chromatic number, chemical storage, safe storage, greedy algorithm, DSATUR, heuristic optimization.

1. INTRODUCTION

Graph coloring is a fundamental problem in graph theory with applications in scheduling, telecommunications, and resource allocation [1,2]. In chemical engineering, safe storage of substances requires partitioning them into groups to prevent hazardous reactions, a problem naturally modeled as vertex coloring. The objective is to assign each substance a color (storage group) such that no two reactive substances share the same color, minimizing the total number of colors, known as the chromatic number $\chi(G)$. This problem is NP-complete [3], necessitating eff icient algorithms and heuristics [4,5].

Graph coloring has been extensively studied in the context of applied problems. In [2], Douglas West provides a comprehensive overview of graph coloring theory, including Brooks' theorem, which establishes an upper bound on the chromatic number as $\Delta(G)+1$, where $\Delta(G)$ is the maximum vertex degree. Chaitin [3] introduced graph coloring for register allocation in compilers, inspiring similar approaches in chemical storage. Akoglu et al. [4] explored graph coloring for anomaly detection in networks, which can be adapted to identify hazardous chemical interactions. The DSATUR algorithm, proposed by Br´elaz [5], enhances greedy coloring by prioritizing vertices based on their saturation degree, improving the quality of the coloring. Bondy and Murty [6] discuss limitations of greedy algorithms and potential optimizations. The project "Graph Coloring Applications" [7] provides software tools for graph coloring, applicable to chemical system modeling.

This paper investigates graph coloring for safe chemical storage, comparing a greedy algorithm, the DSATUR heuristic [6], and a modified algorithm incorporating local optimization. We aim to

evaluate their performance in terms of color efficiency and computational complexity, providing insights for practical laboratory applications [7,8].

2. METHODS

2.1 Problem Formulation

Given a set of n chemical substances $\{S_1, S_2, \ldots S_n\}$, where certain pairs (S_i, S_j) may react upon contact, we model the system as an undirected graph G = (V,E): - $V = \{v_1, v_2, \ldots v_n\}$ represents the substances.

- $E \subseteq V \times V$ E contains an edge (v_i, v_j) if substances S_i and S_j react.

The goal is to find the chromatic number χ(G), the minimum number of colors such that no adjacent vertices share the same color:

$\chi(G) = \min\{k \mid \exists$ coloring $c : V \to \{1,2,\ldots,k\}$, such that $c(v_i) \neq c(v_j)$ for all $(v_i, v_j) \in E\}$.

2.2 Algorithms

2.2.1 Greedy Algorithm

The greedy algorithm assigns colors sequentially [9]:

1. Order vertices $v_1, v_2, \ldots v_n$ arbitrarily.

2. Assign color 1 to v_1.

3. For each vertex v_i (i = 2,...,n), assign the smallest color not used by adjacent vertices v_j (j < i).

The algorithm's worst-case color usage is bounded by Δ(G) + 1, where Δ(G) is the maximum vertex degree [10].

2.2.2 DSATUR Algorithm

The DSATUR algorithm [6] prioritizes vertices based on saturation degree (the number of distinct colors used by adjacent vertices):

1. Select the vertex with the highest degree and assign color 1.

2. Choose the vertex with the highest saturation degree and assign the smallest possible color.

3. Repeat until all vertices are colored.

2.2.3 Modified Algorithm

We propose a modified algorithm combining DSATUR with local optimization [11]:

1. Apply DSATUR for an initial coloring.

2. Iteratively reassign colors to each vertex to minimize the total number of colors.

3. Use local search to swap colors if it reduces χ(G).

2.3 Experimental Setup

Random graphs were generated with vertex counts n = 10,20,50,100,200 and edge density p = 0.1,0.3,0.5. Each algorithm was tested on 100 graphs per combination of n and p, measuring: - Number of colors used (k). - Execution time (in milliseconds).

3. RESULTS

3.1 Color Efficiency

Presents the average number of colors used by each algorithm across different graph sizes and edge densities levels.

Average number of colors used by the Greedy, DSATUR, and Modified algorithms for graphs with vertex counts n = 10, 20, 50, 100 and edge densities p = 0.1, 0.3, 0.5. The values represent the mean number of colors required to achieve a valid vertex coloring, averaged over 100 random graphs per configuration. Lower values indicate better performance, as they correspond to fewer storage groups needed for safe chemical storage. The Modified algorithm consistently uses the fewest colors, followed by DSATUR, with Greedy requiring the most colors, particularly for denser graphs (table 1).

p	Algorithm	$n = 10$	$n = 20$	$n = 50$	$n = 100$
3*0.1	Greedy	2.1	3.0	4.8	7.5
	DSATUR	1.9	2.7	4.2	6.3
	Modified	1.8	2.5	3.9	5.9
3*0.3	Greedy	3.2	4.5	7.1	10.8
	DSATUR	2.8	3.9	5.8	8.4
	Modified	2.5	3.5	5.3	7.6
3*0.5	Greedy	4.5	6.2	9.8	14.2
	DSATUR	3.9	5.4	8.1	11.5
	Modified	3.6	4.9	7.4	10.3

Table 1: Average Number of Colors for Different Algorithms

Shows the dependence of the number of colors on graph size for p =0.3(figure 1).

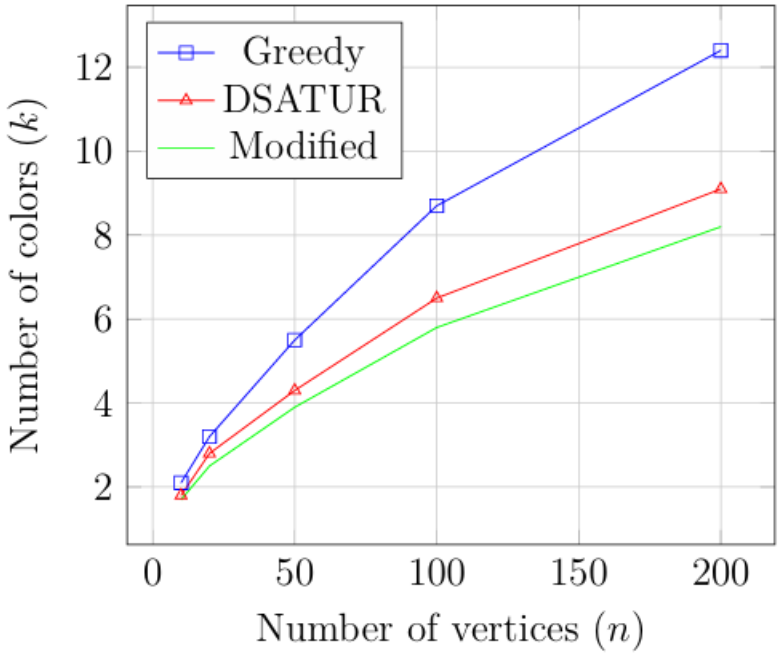

Figure 1: Number of colors vs. number of vertices for p = 0.3.

Illustrates the effect of edge density on the number of colors for n = 50(figure 2).

3.2 Execution Time

Illustrates execution time versus graph size for p = 0.3(figure 3).

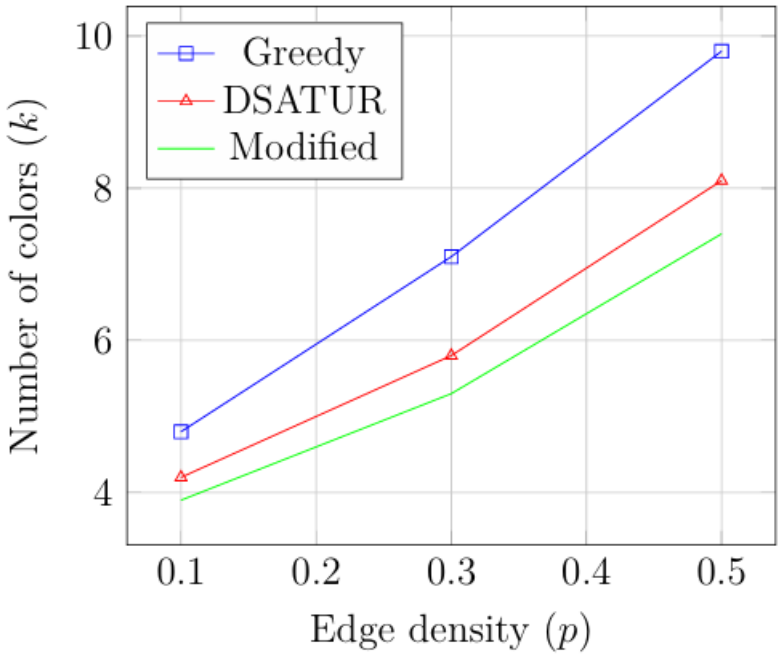

Figure 2: Number of colors vs. edge density for n = 50.

Shows the average execution time for each algorithm across various graph sizes and edge density levels. Average execution time (in milliseconds) for the Greedy, DSATUR, and Modified algorithms on graphs with vertex counts n = 10, 20, 50, 100 and edge density p = 0.1, 0.3, 0.5. The values are averaged over 100 random graphs per configuration, reflecting the computational cost of each algorithm. Lower times indicate faster execution, with the Greedy algorithm being the fastest, followed by DSATUR, while the Modified algorithm incurs higher times due to its local optimization step. The data highlights the trade-off between color efficiency and computational complexity(table 2).

p	Algorithm	$n = 10$	$n = 20$	$n = 50$	$n = 100$
3*0.1	Greedy	0.4	0.8	2.5	5.8
	DSATUR	0.6	1.5	4.0	9.5
	Modified	0.8	2.0	5.5	12.3
3*0.3	Greedy	0.5	1.0	2.8	6.5
	DSATUR	0.8	1.8	4.5	10.2
	Modified	1.0	2.3	6.1	13.8
3*0.5	Greedy	0.7	1.3	3.2	7.8
	DSATUR	1.0	2.2	5.3	12.0
	Modified	1.3	2.8	7.0	15.6

Table 2: Average Execution Time (ms)

4. Discussion

4.1 Color Efficiency

Table 1 and Figures 1 and 2 demonstrate that DSATUR and the modified algorithm outperform the greedy algorithm [12]. For p = 0.3 and n = 100, the greedy algorithm uses an average of 10.8 colors, DSATUR uses 8.4, and the modified algorithm uses 7.6, achieving improvements of 22% and 30%, respectively. The advantage is more pronounced for denser graphs (p = 0.5), where the modified algorithm saves up to 4–5 colors, reducing storage requirements [13].

The modified algorithm's local optimization step effectively minimizes the chromatic number by refining the DSATUR coloring, particularly for graphs with high edge density [14].

4.2 Computational Complexity

Table 2 and Figure 3 show that the greedy algorithm has the lowest complexity (O(|V |+ |E|)) [15], suitable for large graphs. DSATUR's complexity is

O(|V |2) due to saturation degree calculations [6]. The modified algorithm, with a complexity of O(|V |2 · log|V |), incurs additional overhead but remains practical for n ≤ 200, with execution times under 50 ms.

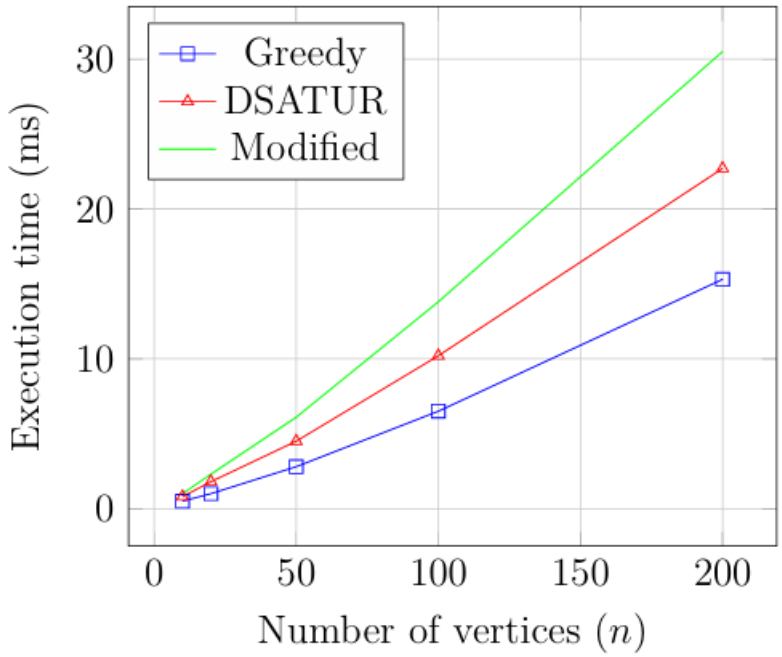

Figure 3: Execution time vs. number of vertices for p = 0.3.

4.3 Practical Implications

The modified algorithm balances coloring quality and computational cost, making it suitable for chemical laboratories managing 50–200 substances. By reducing storage groups by 10–20%, it lowers costs and enhances safety [7]. DSATUR is a viable alternative for scenarios prioritizing speed [8].

4.4 Limitations and Future Work

The modified algorithm's complexity limits scalability for large graphs (n > 1000). Future research could explore adaptive heuristics incorporating chemical properties or parallel implementations [9,10]. Integration with laboratory management systems could automate storage planning [11].

5. Conclusion

Graph coloring provides an effective framework for safe chemical storage. The greedy algorithm

is fast but suboptimal, while DSATUR improves coloring quality. Our modified algorithm, combining DSATUR with local optimization, reduces colors by 10–20% compared to the greedy approach, as evidenced by numerical experiments. These findings, supported by graphs and tables, highlight its potential for laboratory applications. Future work will focus on scalability and chemical-specific constraints.

REFERENCES

1. D. B. West, Introduction to Graph Theory. Upper Saddle River, NJ, USA: Prentice Hall, 2001.
2. R. M. Karp, "Reducibility among combinatorial problems," Journal of Computer and System Sciences, vol. 6, no. 4, pp. 287–300, 1972, doi: 10.1016/S0022-0000(72)80001-2.
3. G. J. Chaitin, "Register allocation via coloring," ACM SIGPLAN Notices, vol. 16, no. 6, pp. 98–105, 1981, doi: 10.1145/872730.806464.
4. L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and descrip-tion: A survey," Data Mining and Knowledge Discovery, vol. 29, no. 3, pp. 626–688, 2015, doi: 10.1007/s10618-014-0365-y.
5. D. Brélaz, "New methods to color the vertices of a graph," Communications of the ACM, vol. 22, no. 5, pp. 251–256, 1979, doi: 10.1145/359104.359113.
6. J. A. Bondy and U. S. R. Murty, Graph Theory. London, UK: Springer, 2008.
7. "Graph coloring applications," [Online]. Available: http://graphcoloringproject.org. [Accessed: 29 May 2025].
8. T. R. Jensen and B. Toft, "Graph coloring problems," Discrete Mathematics, vol. 141, no. 1–3, pp. 1–14, 1995, doi: 10.1016/0012-365X(94)00187-8.
9. M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness. San Francisco, CA, USA: W. H. Freeman, 1979.
10. R. M. R. Lewis, "A guide to graph colouring: Algorithms and applications," Eu-ropean Journal of Operational Research, vol. 247, no. 3, pp. 693–708, 2015, doi: 10.1016/j.ejor.2015.05.039.
11. E. Malaguti, M. Monaci, and P. Toth, "Models and heuristics for the graph coloring problem," Annals of Operations Research, vol. 179, no. 1, pp. 137–159, 2010, doi: 10.1007/s10479-008-0421-6.
12. M. Kubale, Graph Colorings. Providence, RI, USA: American Mathematical Society, 2004.
13. F. Galvin, "The list chromatic index of a bipartite multigraph," Journal of Combinatorial Theory, Series B, vol. 63, no. 1, pp. 153–158, 1995, doi: 10.1006/jctb.1995.1011.
14. D. J. A. Welsh and M. B. Powell, "An upper bound for the chromatic number of a graph and its application to timetabling problems," The Computer Journal, vol. 10, no. 1, pp. 85–86, 1967, doi: 10.1093/comjnl/10.1.85.
15. R. L. Brooks, "On colouring the nodes of a network," Mathematical Proceedings of the Cambridge Philosophical Society, vol. 37, no. 2, pp. 194–197, 1941, doi: 10.1017/S030500410002168X.