Project Proposal: Graph Coloring

Project Number: 10

Team Members:

- Aakarsh Mishra (2024111007)
- Abhijit Suhas (2024101064)
- Rachit Mehta (2024101089)
- Jenik Gajera (2024113026)
- Divyanshu Giri (2024114009)

Title

Graph Coloring: An Analysis of the Trade-off Between Heuristics and Optimality

---

Introduction

The *Graph Coloring Problem* is a classic NP-hard challenge in computer science that asks for the minimum number of colors required to color the vertices of a graph so that no two adjacent vertices share the same color. This minimum number is known as the *chromatic number*.

This project aims to explore the balance between **speed and accuracy** in solving this problem. While finding the exact chromatic number is computationally expensive and impractical for large graphs, heuristic methods can provide near-optimal results efficiently. Our goal is to implement, analyze, and visualize a range of algorithms—from simple heuristics to an exact solver—to highlight this fundamental trade-off.

Algorithms to be Implemented and Compared

We will implement and analyze four distinct algorithms, each representing a different point along the heuristic-to-optimal spectrum:

1. **Welsh–Powell (Simple Greedy Heuristic):** A basic greedy algorithm that sorts vertices by decreasing degree and assigns the smallest available color that does not conflict with adjacent vertices. It is fast but can be suboptimal.
2. **DSatur (Adaptive Heuristic):** An enhancement of the greedy approach that dynamically chooses the next vertex to color based on its *saturation degree*—the number of distinct colors among its neighbors. This often produces better results than the simple greedy method.
3. **Simulated Annealing (Metaheuristic):** A probabilistic search algorithm that begins with a random coloring and iteratively improves it by making small random changes. It occasionally accepts worse solutions to escape local optima, guided by a temperature parameter that decreases over time.
4. **Exact Solver (Dynamic Programming over Subsets):** A method that guarantees the optimal solution by exhaustively exploring partitions of vertices into independent sets. It is exponential in time complexity but serves as a performance and quality baseline for smaller graphs.

Methodology for Analysis and Comparison

The algorithms will be compared empirically based on **solution quality** and **performance**.

1. Test Data Generation

We will construct a diverse set of graphs with varying:

- **Size:** Number of vertices and edges.
- **Density:** Sparse and dense configurations.
- **Structure:** Standard graph types such as planar, bipartite, and random graphs.

2. Evaluation Metrics

- **Solution Quality:** Number of colors used (k), compared to the optimal chromatic number (for small graphs).
- **Runtime:** Execution time (in milliseconds), measured under consistent hardware conditions for fairness.

3. Visualization and Results

- **Performance Plots:** Graphs of runtime vs. number of vertices and solution quality vs. number of vertices to visualize scalability and trade-offs.
- **Graph Visualizer:** An interactive visual tool to display the colorings produced by each algorithm, providing an intuitive understanding of their results.

Stretch Goals (If Time Permits)

1. Real-World Application – University Timetabling

We aim to demonstrate a practical use case of graph coloring in **university exam scheduling**. In this model:

- Each vertex represents a course.
- An edge connects two courses if at least one student is enrolled in both.
- Colors represent distinct exam timeslots.

Using this formulation, our algorithms can generate feasible and conflict-free exam timetables, showcasing their real-world relevance.

2. Theoretical Connection – Maximum Clique Problem

As an additional stretch goal, we may explore the theoretical relationship between the **Graph Coloring Problem** and the **Maximum Clique Problem**. Since the chromatic number of a graph is always at least the size of its largest clique, we plan to:

- Implement a simple recursive algorithm to find the maximum clique.
- Compare the clique size (lower bound) to the chromatic number estimates from our algorithms.

This analysis would provide deeper theoretical insight into graph structure and algorithmic performance.

Conclusion

Through this project, we aim to present a systematic comparison of algorithms that range from heuristic-based to exact solutions for the graph coloring problem. By combining empirical evaluation with visualization, we intend to illustrate the essential trade-off between computational efficiency and solution optimality, offering both practical and theoretical perspectives on one of computer science's most fundamental NP-hard problems.
