**Graph Coloring**

Done by G.Kavitha, CSE BTech IITM

under the guidance of Prof.Rupesh Nasre

Vertices: Numbered from 0 to V-1

Colours used: Numbers 1,2,3,...

1. **Simple Greedy Algorithm**: This algorithm uses Adjacency List representation of the graph. In this algorithm, the vertices are chosen one by one. The lowest possible colour that is not assigned to any of its neighbours is found and this colour is assigned to this vertex.
2. **Welsh-Powell Algorithm (d+1 Algorithm):** This algorithm uses Adjacency List representation of graph. The vertices are sorted based on their degree in descending order. Vertices are chosen one by one in this order and the minimum possible colour that is not assigned to any of the neighbours of this vertex is assigned to this vertex. This algorithm uses fewer colours than the greedy algorithm. This algorithm takes O(V+E) time.
3. **DSATUR Algorithm**: This algorithm also uses Adjacency List representation of graph. In this algorithm, vertices are sorted based on their saturation degree which is the number of neighbours which have already been coloured. The vertex with the highest number of coloured neighbours is chosen to be coloured next. In case of a tie, the vertex with higher degree is chosen. The number of colours used here is lesser than the number used with Welsh Powell Algorithm, but the time take is considerably higher.
4. **DSATUR modified Algorithm:** This is a slightly modified version of DSATUR Algorithm. Here, the vertices are sorted not based on number of coloured neighbours, but based on the number of unique colours among the neighbours of this vertex. In case of a tie, the vertex with the higher degree is chosen. This algorithm works better than DSATUR in terms of the number of colours used, but takes more time than DSATUR.
5. **Backtracking Algorithm:** This algorithm uses Adjacency Matrix representation of the graph. Here the vertices are assigned with colours and if there is a conflict in the assignment of colours, we backtrack and assign the next possible colour to the previous vertex. This way we can guarantee a solution will be found. However, this algorithm has a worst case exponential running time.
6. **Greedy Colouring using the prefix codes**: This algorithm uses the vertex codes and edge codes. The vertex codes are generated based on Huffmann Prefix codes with degrees of the vertices used in place of frequencies. The edge\_code between two vertices is generated by the path between the two vertices in the Huffmann prefix tree. Using this encoding, the vertex with the largest degree (the shortest vertex\_code) is chosen and the minimum possible colour that is not used by any of its neighbours is found. This is done by generating edge\_codes between the current vertex and a different vertex, and if the edge\_code is present in the list of edge\_codes, then that vertex is a neighbour of the current vertex and its colour is noted. This way, the minimum acceptable colour is found out and assigned to the current vertex. This has a running time of O(|V|^2).
7. **Backtracking Algorithm using Prefix Codes:** This algorithm also uses vertex codes and prefix codes generated based on Huffmann Prefix Codes method. In this method, a maximum number of colours that can be used is specified by the user. The colours are assigned one by one to the vertices and if there is a conflict with no possible assignment of colours from 1 to the number specified by the user, then we backtrack and assign the next possible colour to the previous colour. This has worst case exponential running time.