Chromatic Equivalence Classes of Graph

Mahek Parekh
Institute of Technology
Nirma University
Ahmedabad, India
19bce154@nirmauni.ac.in

ABSTRACT

This paper is an introduction to the Chromatic Equivalence Classes of the Graph. Chromatic Equivalence Classes of the Graph is defined and also Chromatic polynomials are defined, their properties are derived, and some practical methods for computing them are given.

The value of the chromatic polynomial of a graph with n vertices gives the number of ways of properly colouring the graph, using λ or fewer colours. Non-isomorphic graphs may possess the same chromatic polynomial. Two graphs G and H are said to be chromatically equivalent, written as G ~ H, if their chromatic polynomials are equal.

KEYWORDS

Chromatic Equivalence Classes of Graph, Chromatic Polynomial, Proper Colouring of Graphs

INTRODUCTION

A colouring of a graph is the result of giving to each node of the graph one of a specified set of colours. In more mathematical terms it is a mapping of the nodes into a specified finite set C which is the set of colours.

By a proper colouring of a graph will be meant a colouring which satisfies the restriction that adjacent nodes are not given i.e., mapped onto the same colour of C. A colouring for which this is not true will be called an improper colouring.

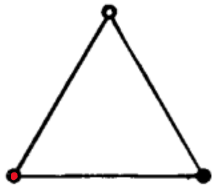
We shall nearly always be concerned with proper colourings only, and it will therefore be convenient to drop the term "proper" and agree that by "colourings" of a graph we mean "proper colourings" unless the contrary is stated.

Graph - G

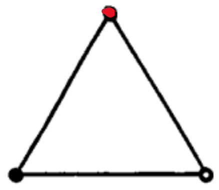
Number of Colours - λ

Function which expresses number of different ways of colouring Graph G with the help of λ colours – $M_G(\lambda)$

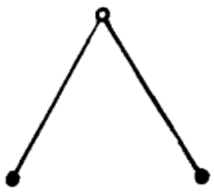
$M_G(\lambda)$ is the number of ways of colouring the Graph G with λ colours, with no stipulation that all the λ colours are in fact used. Hence, it is into mapping.



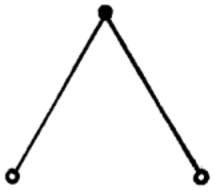
and



We shall regard our graphs as if their nodes were points fixed in space. Hence, colourings of above both triangles will be treated as different.



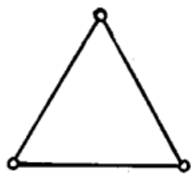
and



We shall take account of different colours. Hence, colouring of above two graphs are considered as different.

SOME ILLUSTRATIONS

We can colour the centre node in any of the λ colours. When this has been done, this colour is no longer available for colouring the outer nodes, by the condition for a proper colouring. Hence the outer nodes can be coloured independently each in λ-1 ways. Thus, $M_G(\lambda) = \lambda(\lambda-1)^2$.

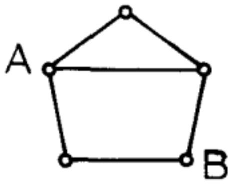


There are λ ways of colouring, say, the top node. There are then λ-1 ways of colouring an adjacent node, and λ-2 ways of colouring the remaining node, since no two nodes may be given the same colour. Thus $M_G(\lambda) = \lambda(\lambda-1)(\lambda-2)$.

This can clearly be generalized. Suppose G is the complete graph on n nodes. We choose a node and colour it; this is possible in λ ways. Picking another node, we have λ-1 colours with which it can be coloured, since it is adjacent to the first node. Pick another node; it is adjacent to both nodes already coloured and can therefore be coloured in λ-2 ways. We continue in this way; the last node can be given any of the remaining λ-(n-1) colours. Hence $M_G(\lambda) = \lambda(\lambda-1)(\lambda-2)\ldots(\lambda-n+1)$. We shall use the notation $\lambda^{(n)}$ for this factorial expression.

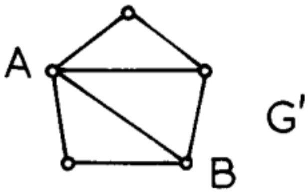
Finally let G be the empty graph on n nodes, i.e., the graph having no edges. Its n isolated nodes can be coloured independently, each in λ ways. Hence for this graph $M_G(\lambda) = \lambda^n$.

A FUNDAMENTAL THEOREM

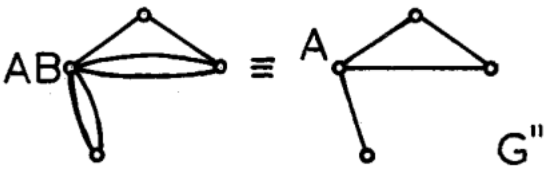


Let us consider a particular graph G, such as that in the above figure, and concentrate on a particular pair of non-adjacent nodes, for example those marked A and B in the figure. Now the colourings of G in λ colours are of two types:

- those in which A and B are given different colours, and
- those in which A and B are given the same colour.



A colouring of G of type (i) will be a colouring of the graph G' obtained from G by adding the edge AB as in figure above, since the addition of this edge does not infringe the requirements for a proper colouring.



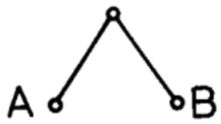
Further, a colouring of G of type (ii) will be a colouring of the graph G" obtained from G by combining the nodes A and B as they are given same colours.

From these two results we derive,

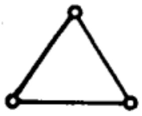
THEOREM 1. $M_G(\lambda) = M_{G'}(\lambda) + M_{G''}(\lambda)$ - a theorem of fundamental importance.

Example:

G =



Hence, G’ =

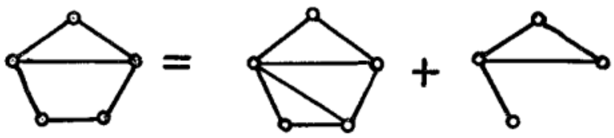


and G’’ =

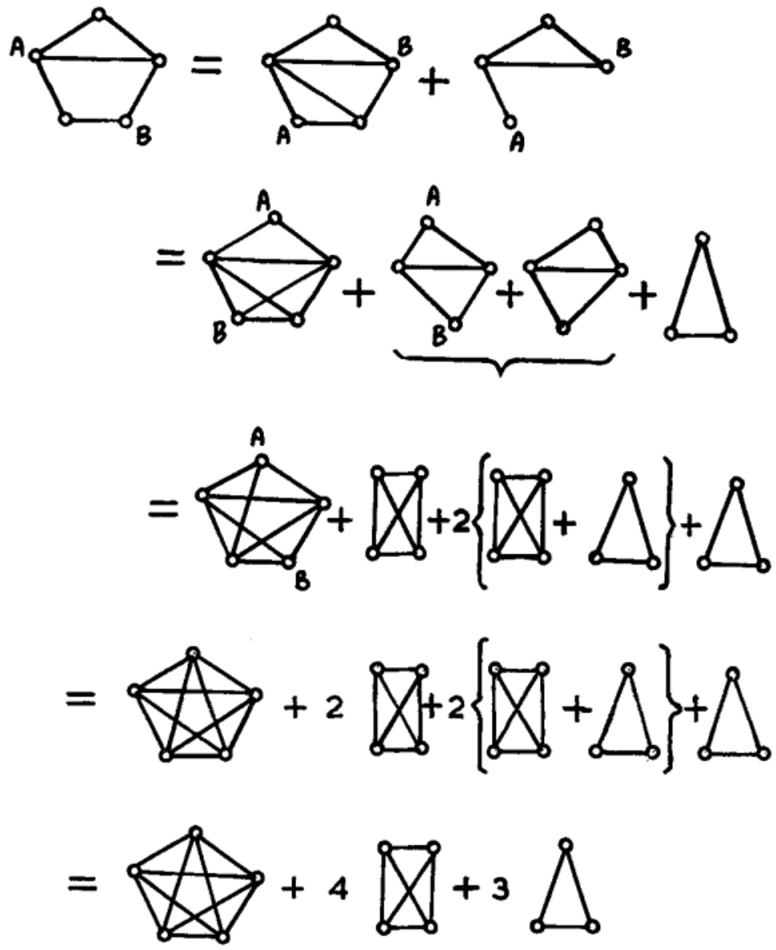
We have already seen that $M_G(\lambda) = \lambda(\lambda-1)^2$, $M_{G'}(\lambda) = \lambda(\lambda-1)(\lambda-2)$ and $M_{G''}(\lambda)=\lambda(\lambda-1)$.

Thus the theorem is verified in this particular case.

To do this in practice it is convenient to adopt a convention whereby the actual picture of a graph serves to denote its chromatic polynomial. Thus instead of writing $M_G(\lambda) = M_{G'}(\lambda) + M_{G''}(\lambda)$, and having to explain what G, G', and G" stand for, we can simply write



Applying Theorem 1 repeatedly to this graph, and indicating by A and B the nodes being considered at each stage, we have

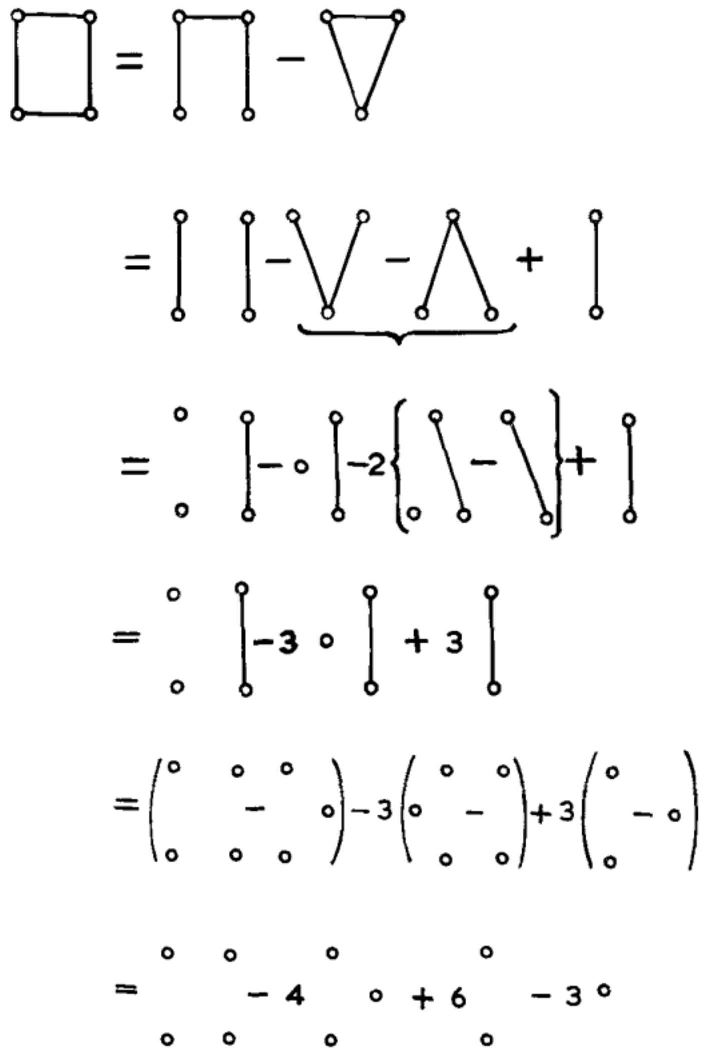


Hence, $M_G(\lambda) = \lambda^{(5)} + 4\lambda^{(4)} + 3\lambda^{(3)}$

When a chromatic polynomial is expressed in this way, we shall say that it is in "factorial form".

Theorem 1 can also be used round the other way, in the form $M_{G'}(\lambda) = M_G(\lambda) - M_{G''}(\lambda)$. Here the process is that of removing edges, and we end up with our chromatic polynomial expressed in terms of the chromatic polynomials of empty graphs.

Thus,



So that, $M_G(\lambda) = \lambda^4$-4 λ^3+6 λ^2-3 λ.

We shall call this process of expressing chromatic polynomials in terms of chromatic polynomials of complete or empty graphs "chromatic reduction."

If a graph G has connected components Ga, G2 ..... Gk, then

$$M_G(\lambda) = M_{G1}(\lambda).\ M_{G2}(\lambda)\ldots M_{Gk}(\lambda).$$

Proof: Since the components are disjoint, the colouring of each is quite independent of the colouring of the others. Hence the number of ways of colouring the whole graph is simply the product of the numbers of colourings of the separate components.

CHROMATIC EQUIVALENCE CLASSES OF A GRAPH

Two graphs G and H are said to be chromatically equivalent, written as G ~ H, if their chromatic polynomials are equal. Non-isomorphic graphs may possess the same chromatic polynomial. A graph G is said to be chromatically unique if no other graph has same Chromatic Polynomial as that of G.

Let G and H be two chromatic equivalent graphs. Then

(i) $v(G) = v(H)$
(ii) $e(G) = e(H)$
(iii) $n_G(C3) = n_H(C3)$
(iv) $c(G) = c(H)$
(v) $b(G) = b(H)$
(vi) G is connected if and only if H is connected
(vii) G is 2-connected if and only if H is 2-connected
(viii) $g(G) = g(H)$
(ix) G is bipartite if and only if H is bipartite

C++ CODE FOR FINDING WHETHER 2 GRAPHS ARE IN SAME EQUIVALENCE CLASS OR NOT

```
#include <bits/stdc++.h>
using namespace std;

//recursive function which gives values of chromatic polynomial of the graph
void chropoly(vector<vector<int>> adjmat, vector<int>& coef)
{
    for(int i=0;i<adjmat.size();i++)
    {
        for(int j=0;j<i;j++)
        {
            if(adjmat[i][j]==0 && i!=j)
            {
                vector<vector<int>>n=adjmat;
                n[i][j]=1;
                n[j][i]=1;
                chropoly(n,coef);
                vector<vector<int>>nn=adjmat;
                for(int k=0;k<adjmat.size();k++)
                {
                    if(nn[i][k]!=1 && nn[j][k]==1)
                    {
                        nn[i][k]=1;
                        nn[k][i]=1;
                    }
                }
                nn.erase(nn.begin()+j);
                for(int i=0;i<nn.size();i++)
                {
                    nn[i].erase(nn[i].begin()+j);
                }
                chropoly(nn,coef);
                return;
```

```
            }
        }
    }
    coef[adjmat.size()]++;
}

int main()
{
    //taking inputs for 1st graph
    int n;
    cout<<"Enter number of vertices in 1st graph: ";
    cin>>n;
    vector<vector<int>> adjmat1(n, vector<int>(n,0));
    cout<<"Enter number of edges in 1st graph: ";
    int e;
    cin>>e;
    cout<<"Enter pair of vertices forming edges: \n";
    int x,y;
    for(int i=0;i<e;i++)
    {
        cin>>x>>y;
        adjmat1[x][y]=1;
        adjmat1[y][x]=1;
    }

    //taking inputs for 2nd graph
    int n2;
    cout<<"Enter number of vertices in 2nd graph: ";
    cin>>n2;
    vector<vector<int>> adjmat2(n2, vector<int>(n2,0));
    cout<<"Enter number of edges in 2nd graph: ";
    int e2;
    cin>>e2;
    cout<<"Enter pair of vertices forming edges: \n";
    for(int i=0;i<e2;i++)
    {
        cin>>x>>y;
        adjmat2[x][y]=1;
        adjmat2[y][x]=1;
    }

    //if number of vertices and edges are not same then declare not in same
equivalent class
    if(n!=n2 || e!=e2)
    {
        cout<<"Two graphs are not in same chromatic equivalence class.\n";
        return 0;
    }
```

```cpp
    vector<int> coef1(n+1,0);
    vector<int> coef2(n+1,0);

    //callinf function chropoly to find chromatic polynomial of the graphs
    chropoly(adjmat1,coef1);
    chropoly(adjmat2,coef2);

    //printing chromatic polynimial of graph 1
    cout<<"Chromatic Polynomial for 1st Graph = ";
    for(int i=n;i>2;i--)
    {
        cout<<coef1[i]<<"*lambda("<<i<<") + ";
    }
    cout<<coef1[2]<<"*lambda(2)";
    cout<<"\n";

    //printing chromatic polynomial of graph 2
    cout<<"Chromatic Polynomial for 2nd Graph = ";
    for(int i=n;i>2;i--)
    {
        cout<<coef2[i]<<"*lambda("<<i<<") + ";
    }
    cout<<coef2[2]<<"*lambda(2)";
    cout<<"\n";

    //declaring whether 2 graphs are in same equivalence class or not
    if(coef1==coef2)
    {
        cout<<"Two graphs are in same chromatic equivalence class.\n";
    }
    else
    {
        cout<<"Two graphs are not in same chromatic equivalence class.\n";
    }

    return 0;
}
```

SAMPLE INPUT AND OUTPUT:

```
PS C:\Users\KETAN PAREKH\Desktop\MAHEK\Semester 6\Graph Theory> cd "c:\Users\K
assigncode.cpp -o assigncode } ; if ($?) { .\assigncode }
Enter number of vertices in 1st graph: 4
Enter number of edges in 1st graph: 3
Enter pair of vertices forming edges:
0 1
0 2
0 3
Enter number of vertices in 2nd graph: 4
Enter number of edges in 2nd graph: 3
Enter pair of vertices forming edges:
0 1
1 2
2 3
Chromatic Polynomial for 1st Graph = 1*lambda(4) + 3*lambda(3) + 1*lambda(2)
Chromatic Polynomial for 2nd Graph = 1*lambda(4) + 3*lambda(3) + 1*lambda(2)
Two graphs are in same chromatic equivalence class.
PS C:\Users\KETAN PAREKH\Desktop\MAHEK\Semester 6\Graph Theory>
```

REFERENCES:

1. Ronald C Read. “An Introduction to Chromatic Polynomials”
2. G L Chia. “On the chromatic equivalence class of graphs”
3. Seyedfakhredin Musavishavazi . “The Chromatic Polynomial of Graphs”
4. B.R. Srinivas , A. Sri Krishna Chaitanya. “The Chromatic Polynomials and its Algebraic Properties”
5. Cody Fout. “The Chromatic Polynomial”
6. Julie Zhang. “An Introduction to Chromatic Polynomials”
7. Amanda Aydelotte “An Exploration of the Chromatic Polynomial”