Malaya Journal of Matematik, Vol. 8, No. 2, 616-621, 2020

https://doi.org/10.26637/MJM0802/0050

C++ Programme for total dominator chromatic number of cycles using elementary transformations

A. Vijayalekshmi[1]* and J. Virgin Alangara Sheeba[2]

Abstract

A total dominator coloring of a graph $G=(V,E)$ without isolated vertices is a proper coloring together with each vertex in G properly dominates a color class. The total dominator chromatic number of G is the minimum number of color classes with additional condition that each vertex in G properly dominates a color class and is denoted by $\chi_{td}(G)$. In this paper, we find the total dominator chromatic number of cycles using elementary transformations through C++ programme.

Keywords

Coloring, Total dominator coloring, Total dominator chromatic number.

AMS Subject Classification

05C69, 68W25.

[1] *Department of Mathematics, S.T.Hindu College, Nagercoil-629002, Tamil Nadu, India.*
[2] *Research Scholar [Reg. No:11813], Department of Mathematics, S.T.Hindu College, Nagercoil-629002, Tamil Nadu, India.*
[1,2] *Affiliated to Manonmaniam Sundaranar University, Abishekapatti, Tirunelveli-627012, Tamil Nadu, India.*
***Corresponding author**:vijimath.a@gmail.com

Article History: Received **21** February **2020**; Accepted **27** April **2020**

Contents

1. Introduction

In this paper we only consider cycles. Further details in graph theory can be found in F. Harrary [4]. Let $G=(V,E)$ be a graph with minimum degree at least one. A cycle on n vertices denoted by C_n is a connected graph where each vertex has degree two. We label the vertices of C_n as v_i for $1 \leq i \leq n$ and let (v_i,v_{i+1})be an edge of C_n.

A proper coloring of G is an assignment of colors to the vertices of G, such that adjacent vertices have different colors. The smallest number of colors for which there exists a proper coloring of G is called a chromatic number of G, and is denoted by $\chi(G)$. A total dominator coloring (td-coloring) of G is a proper coloring of G with extra property that every vertex in G properly dominates color class. The total dominator chromatic number is denoted by $\chi_{td}(G)$ and is defined by the minimum number of colors needed in a total dominator coloring of G. This concept was introduced by Vijayalekshmi in [1]. This notion is also referred as a smarandachely k-dominator coloring of $G,(k \geq 1)$ and was introduced by Vijayalekshmi in [2]. For an integer $k \geq 1$, a smarandachely k-dominator coloring of G is a proper coloring of G, such that every vertex in a graph G properly dominates a k color class. The smallest number of colors for which there exists a smarandachely k-dominator coloring of G is called the smarandachely k-dominator chromatic number of G and is denoted by $\chi_{td}^{s}(G)$.

In a proper coloring C of a graph G, a color class of C is a set consisting of all those vertices assigned the same color. Let C be a minimum td-coloring of G. We say that a color class is called a non-dominated color class($n-d$ color class) if it is not dominated by any vertex of G and these color classes are also called repeated color classes.

For more details on this theory and its applications, we suggest the reader to refer [3, 5, 6].

2. Preliminaries

In this section, we recall the crucial theorem [3] which is very useful in our work. The total dominator chromatic number of cycles was found in the following observation.

Let G be C_n. Then

$$\chi_{td}\ (C_n) = \begin{cases} 2\lfloor \frac{n}{4} \rfloor + 2, & if n \equiv 0(mod 4) \\ 2\lfloor \frac{n}{4} \rfloor + 3, & if n \equiv 1(mod 4) \\ 2\lfloor \frac{n+2}{4} \rfloor + 2, & otherwise. \end{cases}$$

In this paper, we obtain a C++ programme to find the td-chromatic number of cycles by using elementary transformations.

3. Main Result

In this section, We have to find the total dominator chromatic number of cycles using C++ programme. The C++ programme is successfully compiled and run on C++ platform. The runtime test is included.

Programme as follows

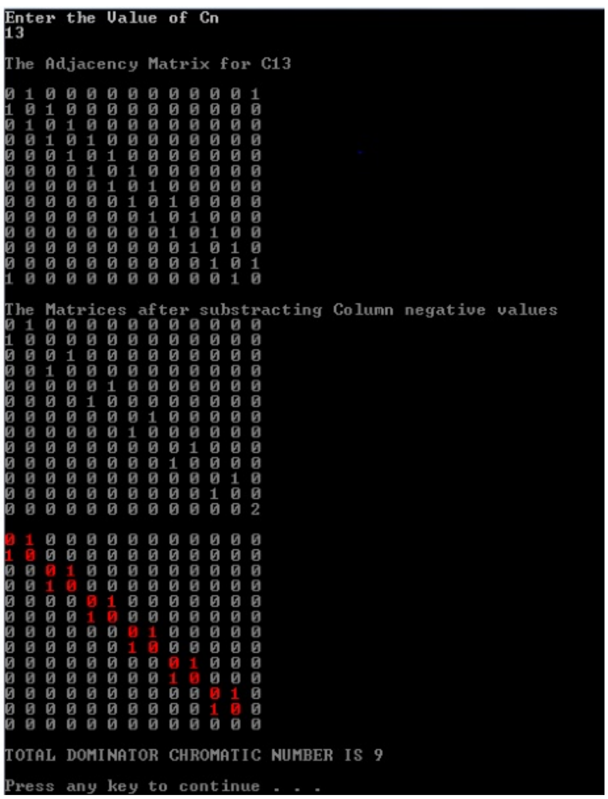
```
#include "stdafx.h"
#include <Windows.h>
#include <conio.h>
#include <iostream>
using namespace std;
int main()
{
int inpt;
cout << "Enter the Value of Cn" << endl;
cin >> inpt;
int N = inpt, M = inpt; int** ary = new int*[N]; int** mat = new int*[N];
 int** mat1 = new int*[N]; int** matsum = new int*[N];
for (int i = 0; i < N; ++i)
{
ary[i] = new int[M], mat[i] = new int[M], mat1[i] = new int[M], matsum[i]
 = new int[M];
}
int k, l, sum;
HANDLE p = GetStdHandle(STD_OUTPUT_HANDLE);
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
for (int i = 0; i < N; ++i)
for (int j = 0; j < M; ++j)
ary[i][j] = i;
cout << "\n" << "The Adjacency Matrix for C" << N << "\n" << "\n";
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (ary[j][i] == i + 1 | ary[j][i] == i - 1 | ary[j][i] == i + (N - 1)
| ary[j][i] == i - (N -1))
{
mat[i][j] = 1;
cout << mat[i][j] << " ";
}
else
{
mat[i][j] = 0;
cout << mat[i][j] << " ";
}
}
cout << "\n";
}
for (int i = 0; i < N; i++)
```

```
{
for (int j = 0; j < N; j++)
{
if (i % 2 == 0)
{
if (i >= 2)
{
mat1[i][j] = mat[i][j] - mat1[i - 2][j];
}
else
{
mat1[i][j] = mat[i][j];
}
}
else
{
mat1[i][j] = mat[i][j];
}
}
}
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (i % 2 == 0)
{
if (mat1[i][j] == 1 && mat1[i][N-1] == 1 && i < N-2 )
{
mat1[i][(N - 1)] = mat1[i][j] - mat1[i][(N - 1)];
}
else if (mat1[i][j] == 1 && mat1[i][N - 1] == -1 && i < N-2)
{
mat1[i][(N - 1)] = mat1[i][j] + mat1[i][(N - 1)];
}
else
{
mat1[i][j] = mat1[i][j];
}
}
else
{
mat1[i][j] = mat[i][j];
}
}
}
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (j % 2 == 0)
{
if (j >= 2)
{
mat1[i][j] = mat1[i][j] - mat1[i][j - 2];
}
else
```

```
{
mat1[i][j] = mat1[i][j];
}
}
else
{
mat1[i][j] = mat1[i][j];
}
}
}
cout << "\n" << "The Matrixes after subtracting Column negative values" << "\n";
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (j % 2 == 0)
{
if (mat1[i][j] == 1 && mat1[N-1][j] == 1)
{
mat1[N-1][j] = mat1[i][j] - mat1[(N-1)][j];
cout << mat1[i][j] << " ";
}
else if (mat1[i][j] == 1 && mat1[(N-1)][j] == -1)
{
mat1[(N - 1)][j] = mat1[i][j] + mat1[(N - 1)][j];
cout << mat1[i][j] << " ";
}
else
{
mat1[i][j] = mat1[i][j];
cout << mat1[i][j] << " ";
}
}
else
{
mat1[i][j] = mat1[i][j];
cout << mat1[i][j] << " ";
}
}
cout << "\n";
}
cout << "\n";
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (mat1[i][j] == 2 || mat1[i][j] == -2)
{
mat1[i][j] = mat1[i][j] / 2;
}
}
}
if (mat1[N - 1][N - 1] == 1 || mat1[N - 1][N - 1] == -1)
{
for (int i = 0; i < N; i++)
{
```

```
if (mat1[N - 1][N - 1] == 1)
{
mat1[N - 1][i] = mat1[N - 1][i] - 1;
}
else
{
mat1[N - 1][i] = mat1[N - 1][i] + 1;
}
}
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (mat1[i][j] == 1 && mat1[N - 1][j] == -1)
{
mat1[N - 1][j] = mat1[N - 1][j] + mat1[i][j];
}
else if (mat1[i][j] == 1 && mat1[N - 1][j] == 1)
{
mat1[N - 1][j] = mat1[N - 1][j] - mat1[i][j];
}
else
{
mat1[i][j] = mat1[i][j];
}
}
}
}
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (j % 2 == 0 && i % 2 == 0 && mat1[i][j] == 0 && mat1[i][j + 1] == 1
 || mat1[i][j] == 1)
{
SetConsoleTextAttribute(p, FOREGROUND_RED | FOREGROUND_INTENSITY);
cout << mat1[i][j] << " ";
}
else if (j % 2 != 0 && i % 2 != 0 && mat1[i][j] == 0 && mat1[i][j - 1] == 1)
{
SetConsoleTextAttribute(p, FOREGROUND_RED | FOREGROUND_INTENSITY);
cout << mat1[i][j] << " ";
}
else
{
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
cout << mat1[i][j] << " ";
}
}
cout << "\n";
}
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
int O = 0;
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
```

```
{
if (i+1 < N && mat1[i][j] == 1 && mat1[i + 1][j - 1] == 1)
{
O = O + 1;
}
}
}
cout << "\n" << "TOTAL DOMINATOR CHROMATIC NUMBER IS " << O + 3 << "\n" << "\n";
system("Pause");
return 0;
for (int i = 0; i < N; ++i)
{
delete[] ary[i], ary, mat1[i], mat1, mat[i], mat, matsum[i], matsum;
}
return 0;
}
```



4. Conclusion

In this manuscript, we find the total dominator chromatic number of cycles using elementary transformations through C++ programme in simplified and improved manner.

References

[1] A. Vijayalekshmi, Total dominator colorings in paths, *International Journal of Mathematical Combinatorics*, 2(2012), 89–95.

[2] A. Vijayalekshmi, Total dominator colorings in cycles, *International Journal of Mathematical Combinatorics*, 4(2012), 92–96.

[3] A. Vijayalekshmi and J.Virgin Alangara Sheeba, Total dominator chromatic number of Paths, Cycles and Ladder graphs, *International Journal of Contemporary Mathematical Sciences*, 13(5)(2018), 199–204.

[4] F. Harrary, *Graph Theory*, Addition-Wesley, Reading Mass, 1969.

[5] M.I. Jinnah and A. Vijayalekshmi, *Total Dominator Colorings in Graphs*, Ph.D Thesis, University of Kerala, 2010.

[6] Terasa W. Haynes, Stephen T. Hedetniemi, Peter J. Slater, *Domination in Graphs*, Marcel Dekker, New York, 1998.

⋆⋆⋆⋆⋆⋆⋆⋆⋆
ISSN(P):2319 − 3786
Malaya Journal of Matematik
ISSN(O):2321 − 5666
⋆⋆⋆⋆⋆⋆⋆⋆⋆