Malaya Journal of Matematik, Vol. 8, No. 4, 1480-1487, 2020

https://doi.org/10.26637/MJM0804/0024

C++ Programme for total dominator chromatic number of ladder graphs through simple transformations

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

Abstract

A total dominator coloring of a graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ without isolated vertices, along with each vertex in $\mathbb{G}$, is a proper coloring that dominates a color class. The total chromatic dominator number of $\mathbb{G}$ is the minimum number of color classes with further assumption that each vertex in $\mathbb{G}$ dominates a color class properly and is represented as $\chi_{td}(\mathbb{G})$. In this manuscript, we consider the chromatic total dominator number of ladder graphs through fundamental transformations via the program C++.

Keywords

Coloring, Total dominator coloring, Total dominator chromatic number.

AMS Subject Classification

05C69, 68W25.

[1] *Research Scholar [Reg. No:11813], Department of Mathematics, S.T. Hindu College, Nagercoil-629002, Tamil Nadu, India.*
[2] *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 **14** March **2020**; Accepted **02** June **2020**

Contents

1. Introduction

We mainly find ladder graphs in this manuscript. For additional information in graph theory and its applications, we suggest the reader to refer F. Harrary [4]. Allow $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ be a graph without isolated vertices. For any two graphs $\mathbb{G}$ and $\mathbb{H}$, we characterize the cartesian product, signified by $\mathbb{G} \times \mathbb{H}$, to be the graph with vertex set $\mathbb{V}(\mathbb{G}) \times \mathbb{V}(\mathbb{H})$ and edges between two vertices (u_1, v_1) and (u_2, v_2) iff either $u_1 = u_2$ and $v_1v_2 \in E(H)$ or $u_1u_2 \in E(G)$ and $v_1 = v_2$.

In general, for $n \geq 2$, we characterize a ladder graph as $P_2 \times P_n$ and is signified by L_n and $|\mathbb{V}(L_n)| = p = 2n, n \geq 2$.

A proper coloring of $\mathbb{G}$ is an assignment of colors to the vertices of $\mathbb{G}$, in a way that adjacent vertices have different colors. The smallest number of colors for which $\mathbb{G}$ is properly colored is considered a chromatic number of $\mathbb{G}$, and $\chi(\mathbb{G})$ is denoted. A total dominator coloring (td-coloring) of $\mathbb{G}$ is a proper coloring of $\mathbb{G}$ with additional axioms that is properly dominated color class by every vertex in $\mathbb{G}$. Let $\chi_{td}(\mathbb{G})$ be the total dominator chromatic number and is defined by the minimum number of colors needed in a total dominator coloring of $\mathbb{G}$. This principle was developed in [1] by Vijayalekshmi. This thought is often pointed to as a $\mathbb{G}, (k \geq 1)$Smarandachely k-dominator color and was presented in [2] by Vijayalekshmi. A Smarandachely k-dominator coloring of $\mathbb{G}$ for an integer $k \geq 1$ is a proper coloring of G, so that each vertex in a $\mathbb{G}$ graph properly dominates a color class of k. The smallest number of colors for which there exists a Smarandachely k-dominator coloring of $\mathbb{G}$ is called the Smarandachely k-dominator chromatic number of $\mathbb{G}$ and is denoted by $\chi^s_{td}(\mathbb{G})$.

Let $\mathscr{C}$ be a minimum td-coloring of $\mathbb{G}$. We say a color class is considered a non-dominated color class ($n-d$ color class) if no vertex of $\mathbb{G}$ dominates it and these color classes are often considered repeated color classes.

We recommend the author to pertain to [3, 5, 6] for further information on this theory and its applications.

2. Preliminaries

In this segment, we remember the critical [3] theorem which is quite helpful in our research. For the subsequent observation the minimum dominator chromatic number of ladder graphs has been identified.

For every $n \geq 2$, the total dominator chromatic number of a ladder graph is

$$\chi_{td}\left(\mathscr{C}_n\right) = \begin{cases} 2\lfloor \frac{p}{6} \rfloor + 2, & if n \equiv 0(mod 6) \\ 2\lfloor \frac{p-2}{6} \rfloor + 4, & \\ 2\lfloor \frac{p-4}{6} \rfloor + 4, & otherwise. \end{cases}$$

In this manuscript we obtain a C++ program which uses fundamental transformations to find the td-chromatic number of ladder graphs.

3. Main Result

In this section, We have to find the total dominator chromatic number of ladder graphs 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 Ln" << endl;
cin >> inpt;
int N = inpt + inpt; int M = inpt + inpt;
int** ary = new int*[N];
int** mat = new int*[N];
int** mat1 = new int*[N];
int** mat2 = new int*[N];
int** mat3 = 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];
mat2[i] = new int[M]; mat3[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 L" << inpt << "\n" << "\n";
for (int i = 0; i < N; i++)
{
if (i % 2 == 0)
{
for (int j = 0; j < N; j++)
{
if (ary[j][i] == i + 1 | ary[j][i] == i - 1 | ary[j][i] == i + 3)
{
mat[i][j] = 1;
cout << mat[i][j] << " ";
}
```

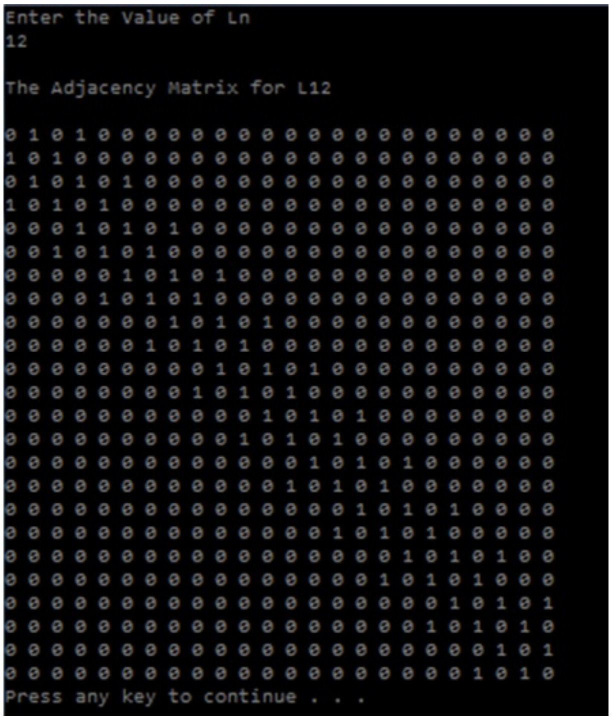
```
else
{
mat[i][j] = 0;
cout << mat[i][j] << " ";
}
}
}
else
{
for (int j = 0; j < N; j++)
{
if (ary[j][i] == i + 1 | ary[j][i] == i - 1 | ary[j][i] == i - 3)
{
mat[i][j] = 1;
cout << mat[i][j] << " ";
}
else
{
mat[i][j] = 0;
cout << mat[i][j] << " ";
}
}
}
cout << "\n";
}
cout << "\n" << "ADJACENCY MATRIX BY SUBSTRATING THE ROW ASSENDING VALUES" << "\n";
for (int i = 0; i < N; i++)
{
int sum = 0;
for (int j = 0; j < N; j++)
{
if (i >= 2 && i <= 5 && mat[i][j] == 1 && mat1[i - 2][j] == 1)
{
mat1[i][j] = mat[i][j] - mat[i - 2][j];
}
else if (i >= 6 && mat[i][j] == 1 && mat1[i - 2][j] == 1 && matsum[i - 4][0] != 1)
{
mat1[i][j] = mat[i][j] - mat[i][j];
}
else if (i >= 4 && mat[i][j] == 1 && mat1[i - 2][j] == 0 && matsum[i - 4][0] == 1)
{
mat1[i][j] = mat[i][j] - mat1[i - 4][j];
}
else
{
mat1[i][j] = mat[i][j];
}
sum = sum + mat1[i][j];
}
matsum[i][0] = sum;
}
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (mat1[i][j] == 1)
```

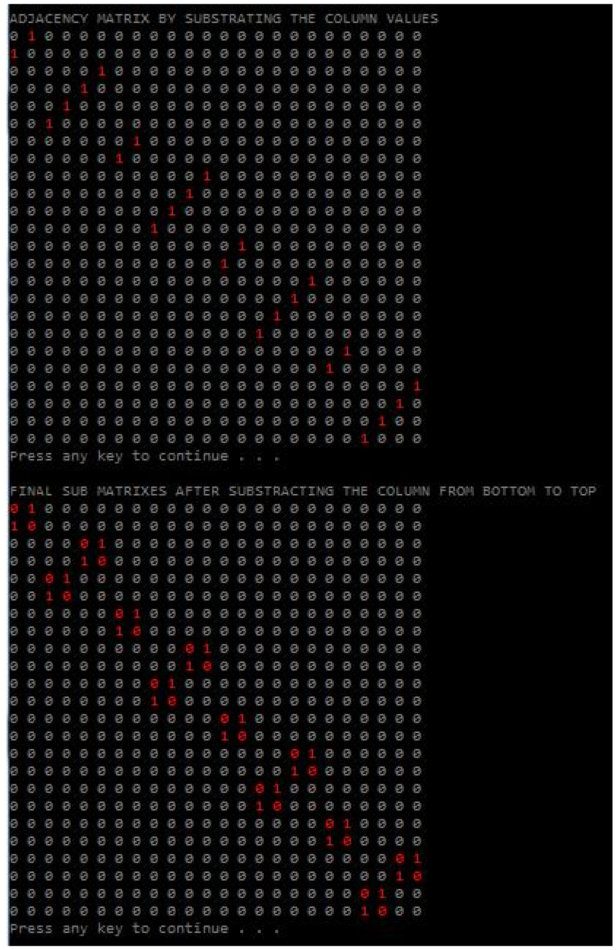
```
{
SetConsoleTextAttribute(p, FOREGROUND_RED | FOREGROUND_INTENSITY);
cout << mat1[i][j] << " ";
}
else
{
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
cout << mat1[i][j] << " ";
}
}
cout << "\n";
}
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
cout << "\n"<< "\n" << "ADJACENCY MATRIX BY SUBSTRATING THE COLUMN VALUES";
if (N%3 == 0)
{
N = N - 1;
for (int i = N; i >= 0; i--)
{
for (int j = N; j >= 0; j--)
{
if (i >= 4 && mat1[i][j] == 1 && mat1[i - 4][j] == 1)
{
mat1[i - 4][j] = mat1[i][j] - mat1[i - 4][j];
}
else if (i >= 2 && mat1[i][j] == 1 && mat1[i - 2][j] == 1)
{
mat1[i - 2][j] = mat1[i][j] - mat1[i - 2][j];
}
}
}
N = N + 1;
cout << "\n";
}
else
{
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (mat1[i][j] == 1 && mat1[i][j+2] == 1)
{
mat1[i][j+2] = mat1[i][j+2] - mat1[i][j];
}
else if (mat1[i][j] == 1 && mat1[i][j + 4] == 1)
{
mat1[i][j + 4] = mat1[i][j + 4] - mat1[i][j];
}
else
{
mat1[i][j] = mat1[i][j];
}
}
}
cout << "\n";
}
```

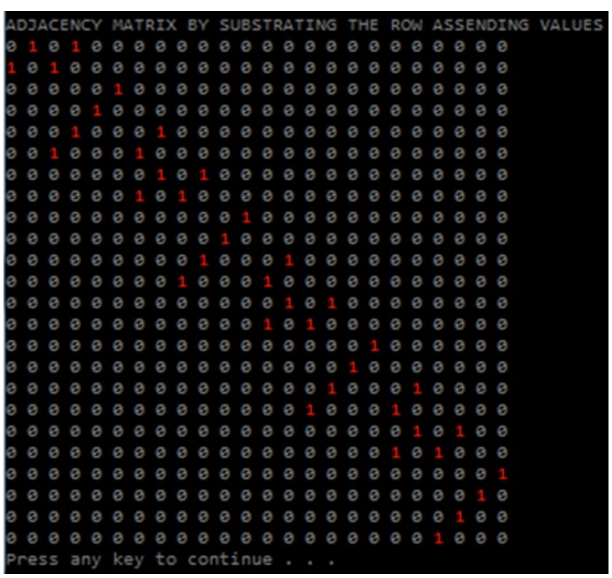
```
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (mat1[i][j] == 1)
{
SetConsoleTextAttribute(p, FOREGROUND_RED | FOREGROUND_INTENSITY);
cout << mat1[i][j] << " ";
}
else
{
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
cout << mat1[i][j] << " ";
}
}
cout << "\n";
}
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
cout << "\n";
int ary2[] = { 0,1,4,5,2,3 }, aaa = 0;
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if (ary2[aaa] > N-1)
{
ary2[aaa] = ary2[aaa]-2;
mat3[i][j] = mat1[ary2[aaa]][j];
}
mat3[i][j] = mat1[ary2[aaa]][j];
}
if (aaa < 5 )
{
ary2[aaa] = ary2[aaa] + 6;
aaa = aaa + 1;
}
else if (aaa = 5)
{
ary2[aaa] = ary2[aaa] + 6;
aaa = 0;
}
}
cout << "FINAL SUB MATRIXES AFTER SUBSTRACTING THE COLUMN FROM BOTTOM TO TOP
" << "\n";
k = 0;
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);
k = k + 1;
cout << mat1[i][j] << " ";
}
else
{
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
cout << mat1[i][j] << " ";
}
}
cout << "\n";
}
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
cout << "\n" << "FINAL SUB MATRIXES AFTER INTERCHANGING THE COLUMN" << "\n";
for (int i = 0; i < N; i++)
{
for (int j = 0; j < N; j++)
{
if ( i% 2==0 && mat3[i][j] == 0 && mat3[i][j + 1] == 1 || mat3[i][j] == 1)
{
SetConsoleTextAttribute(p, FOREGROUND_RED | FOREGROUND_INTENSITY);
cout << mat3[i][j] << " ";
}
else if (i % 2 != 0 && mat3[i][j] == 0 && mat3[i][j - 1] == 1)
{
SetConsoleTextAttribute(p, FOREGROUND_RED | FOREGROUND_INTENSITY);
cout << mat3[i][j] << " ";
}
else
{
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
cout << mat3[i][j] << " ";
}
}
cout << "\n";
SetConsoleTextAttribute(p, FOREGROUND_INTENSITY | FOREGROUND_INTENSITY);
}
cout << "\n";
if (inpt % 3 == 0)
{
cout <<"\n" << "TOTAL DOMINATOR CHROMATIC NUMBER IS " <<  (2 * (k / 3)) + 2
 << "\n";
}
else
{
cout << "\n" << "TOTAL DOMINATOR CHROMATIC NUMBER IS " <<  (2 * (k-1)/3) + 4
 << "\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, mat2[i], mat2,
 mat3[i], mat3;
}
```

```
return 0;
}
```







4. Conclusion

Within this manuscript, we treat the total dominator chromatic number of ladder graphs in a simplified and enhanced fashion utilizing elementary transformations by C++ programme.

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
⋆⋆⋆⋆⋆⋆⋆⋆⋆