Available online at www.sciencedirect.com
ScienceDirect
Future Generation Computer Systems 23 (2007) 716–720
FGCS
FUTURE GENERATION COMPUTER SYSTEMS
www.elsevier.com/locate/fgcs

# Applying *Mathematica* and *webMathematica* to graph coloring

Ünal Ufuktepe[a,*], Goksen Bacak[b]

[a] *Izmir Institute of Technology, Department of Mathematics, Urla, Izmir, Turkey*
[b] *Yasar University, Alsancak, Izmir, Turkey*

Received 1 October 2006; accepted 17 October 2006
Available online 26 December 2006

**Abstract**

This paper analyzes some graph issues by using the symbolic program *Mathematica* and its version for the Web, *webMathematica*. In particular, we consider the problem of graph coloring: the assignment of colors to the vertices/edges of the graph such that adjacent vertices/edges are colored differently. In addition, we address the problem of obtaining the tenacity of binomial trees with *Mathematica*. Finally, we describe briefly an example of the application of our software to a scheduling problem.
© 2006 Elsevier B.V. All rights reserved.

*Keywords:* Graph coloring; Vertex coloring; Edge coloring; Chromatic number; Mathematica

## 1. Introduction

The use of graphs is quite standard in many scientific fields, including paths in networks for grids [7,18], peer-to-peer networks [5,9], neural and functional networks for artificial intelligence [8,10,11,14,17], automatic differentiation [15], interactive optimization [16], text classification [13], etc.

A graph $G$ is a mathematical structure consisting of two sets $V(G)$ (vertices of $G$) and $E(G)$ (edges of $G$). Proper coloring of a graph is an assignment of colors either to the vertices of the graph, or to the edges, in such a way that adjacent vertices/edges are colored differently [1,2,12,21,22]. These problems are usually referred to as *vertex coloring* and *edge coloring* problems, respectively.

Vertex and edge coloring play remarkable roles in many applied problems, including scheduling, clustering, assignment of radio frequencies, separating combustible chemical combinations, and computer optimization. Compiler optimization is a classical application for coloring, where we seek to schedule the use of a finite number of registers. In a program fragment to be optimized, each variable has a range of times during which its value must be kept intact, and in particular, after it is initialized and before its final use. Any two variables whose life spans intersect cannot be placed in the same register. One feasible approach to this problem is to construct a graph where there is a variable associated with each vertex and then to add an edge between any two vertices representing variables whose life spans intersect. A coloring of the vertices of this graph assigns the variables to classes such that two variables with the same color do not clash, and so can be assigned to the same register. No conflicts can occur if each vertex is colored with a distinct color. However, the goal is to find a coloring using the minimum number of colors, because computers have a limited number of registers. The minimum number of colors required for a proper vertex coloring of a graph $G$ is known as its *chromatic number* of $G$, denoted by $\chi(G)$. Similarly, $\chi'(G)$ denotes the *chromatic index* of $G$, that is the minimum number of colors necessary to color the edges of $G$. Graph coloring is an optimization problem: it has been proved that, for a given graph $G$, $\chi'(G)$ is either $\triangle(G)$ or $\triangle(G)+1$, where $\triangle(G)$ denotes the maximum degree of a vertex in $G$. Then the graph $G$ belongs to one of two classes; either to class 1 or to class 2. This classification problem is NP-complete, and this implies that there are no polynomial-time algorithms for this problem [3].

On the other hand, the stability of a communication network consisting of processing nodes and communication links is of prime importance to network designers. As the network begins to lose links or nodes, eventually there is a

* Corresponding author.
*E-mail addresses:* unalufuktepe@iyte.edu.tr (Ü. Ufuktepe), goksen.bacak@yasar.edu.tr (G. Bacak).

loss in its effectiveness. Thus, communication networks must be constructed to be as stable as possible, not only with respect to initial disruptions, but also with respect to the possible reconstruction of the network. One way of measuring the stability of a network (computer, communication, or transportation) is through the ease (or the cost) with which one can disrupt the network. To measure the stability of a network we have some graph parameters like connectivity, integrity, toughness and tenacity. In an analysis of the stability of a network to disruption, these parameters allow us to deal with two fundamental questions about the resulting graph:

(1) How many vertices can still communicate ?
(2) How difficult is it to reconnect the network ?

The *connectivity* is the minimum number of vertices whose removal disconnects the graph. The connectivity gives the minimum cost to disrupt the network but it does not take into account what remains after its disruption. The *integrity* is defined by $I(G) = \min_{S\subset V(G)}\{|S| + m(G - S)\}$, where $m(G - S)$ denotes the order of a largest component of $G - S$ and is determined using ad hoc methods. The connectivity and integrity account for the first question. The second question is addressed by the *toughness* of a graph, denoted by $t(G)$, that is defined as $t(G) = \min\{|S|/w(G - S) : S \subseteq V\}$, where the minimum is taken over all cut-sets $S$ in V(G).

One might, however, handle both of the above questions simultaneously. The basic idea is to associate the cost with the number of vertices destroyed to get small components and the reward with the number of components remaining after destruction. The concept of *tenacity* was introduced by Cozzens, Moazzami and Stueckle [6] as a measure of graph stability in this sense. Formally, the tenacity is $T(G) = \min\{(|S| + m(G/S))/w(G - S) : S \subseteq V\}$, where $m(G - S)$ and $w(G - S)$ denote the order of the largest component and the number of components in $G - S$, respectively. The tenacity for several classes of graphs is analyzed in [4,6].

In this paper we deal with some graph issues by using the symbolic program *Mathematica* and its version for the Internet, *webMathematica*. *Mathematica* is a high-level programming language [25]. It is also a system developed for doing mathematics by computer. *webMathematica* adds interactive calculations and visualizations to a web site by integrating *Mathematica* with the latest web server technology [24]. It makes all of the functionality of *Mathematica* available over the web. The use of *Mathematica* in Graph Theory has been extensively explained by Steven Skiena [19]. His `Combinatorica` package, an extension to the popular computer algebra system *Mathematica*, is the most comprehensive software available for graph theory. It includes functions for constructing graphs and other combinatorial objects, for computing invariants of these objects, and finally for displaying them. It has been perhaps the most widely used software for teaching and research in discrete mathematics since its initial release in 1990 [20].

In this paper, we consider the problem of graph coloring: an assignment of colors to the vertices/edges of the graph such that adjacent vertices/edges are colored differently [23]. In addition, we address the problem of obtaining the tenacity of binomial trees with *Mathematica*. Finally, we describe briefly an example of the application of our software to a scheduling problem.

## 2. *Mathematica* and *webMathematica* applications

In this section, we show some applications of our `ColorG` package to coloring the vertices and the edges of graphs on the web by using *webMathematica*. Interested readers are kindly invited to visit the URL: http://gauss.iyte.edu.tr:8080/webMathematica/goksen/index.msp to run these and many other examples by themselves. We use some commands in the `Combinatorica` package with *Mathematica* to color the vertices/edges of graphs and to give web-based examples with *webMathematica*. Similarly, we use some commands in the `ColorG` package with *Mathematica* to color the edges of graphs and to give web-based examples with *webMathematica*. In this example, the user can select the `CompleteGraph`, `Cycle`, `Grid Graph`, `Star`, `RandomTree` of any graph and enter the number of vertices to get the vertex-colored graph. If the user presses the `Grid Graph` button and enters the number of rows and columns, he/she can get the vertex-colored grid graph as in Fig. 1.

The most important operations on graphs are the sum, union, join, and product of two graphs. `ColorG` does these operations with the most common graphs: complete graph, random tree, wheel, and cycle with *webMathematica* as shown in Fig. 2.

If the number of the vertices of a graph is entered, it is possible to see the Eulerian/Hamiltonian cycle in that graph if one exists. Interested readers are kindly invited to try this at the URL: http://gauss.iyte.edu.tr:8080/webMathematica/atina/index1.msp.

### 2.1. An application

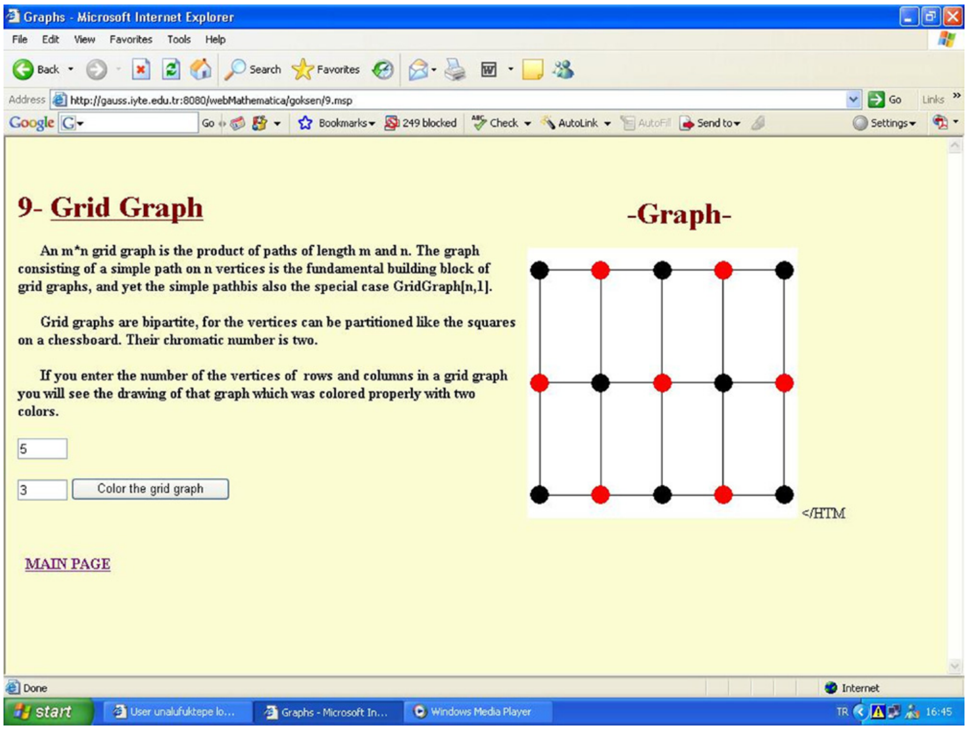
Some scheduling problems induce a graph coloring, i.e., an assignment of positive integers (colors) to the vertices of a graph. We discuss a simple example for coloring the vertices of a graph with a small number $k$ of colors and present computational results for calculating its chromatic number, i.e., the minimal possible value of such a $k$. There are seven tour bus companies in the Istanbul area, each visiting no more than three different locations from among Taksim, Kadikoy, Bakirkoy, and Sultanahmet during a day. The same location cannot be visited by more than one tour bus company on the same day.

If we set up a schedule so that every bus is used once, and every place is visited at least once, how many days will the schedule take ?

Set of tour bus companies: $B_1, B_2, B_3, B_4, B_5, B_6, B_7$.

Locations for each company: {Taksim}, {Taksim, Kadikoy}, {Bakirkoy}, {Kadikoy, Bakirkoy}, {Taksim, Sultanahmet}, {Bakirkoy, Sultanahmet}, {Kadikoy, Sultanahmet}.

Let $B = \{1, 2, 3, 4, 5, 6, 7\}$ be a set of tour bus companies and $P = \{1, 2, 3, 4\}$ be the set of locations (respectively Taksim, Kadikoy, Bakirkoy, Sultanahmet). $P(b)$ be the set of locations which will be visited by the bus company $b \in B$ (Fig. 3). Form a graph $G = G(B, E)$, where $x, y \in B$ are


Graphs - Microsoft Internet Explorer
http://gauss.iyte.edu.tr:8080/webMathematica/goksen/9.msp
-Graph-
Color the grid graph
MAIN PAGE
</HTM

9- Grid Graph

An m*n grid graph is the product of paths of length m and n. The graph consisting of a simple path on n vertices is the fundamental building block of grid graphs, and yet the simple pathbis also the special case GridGraph[n,1].

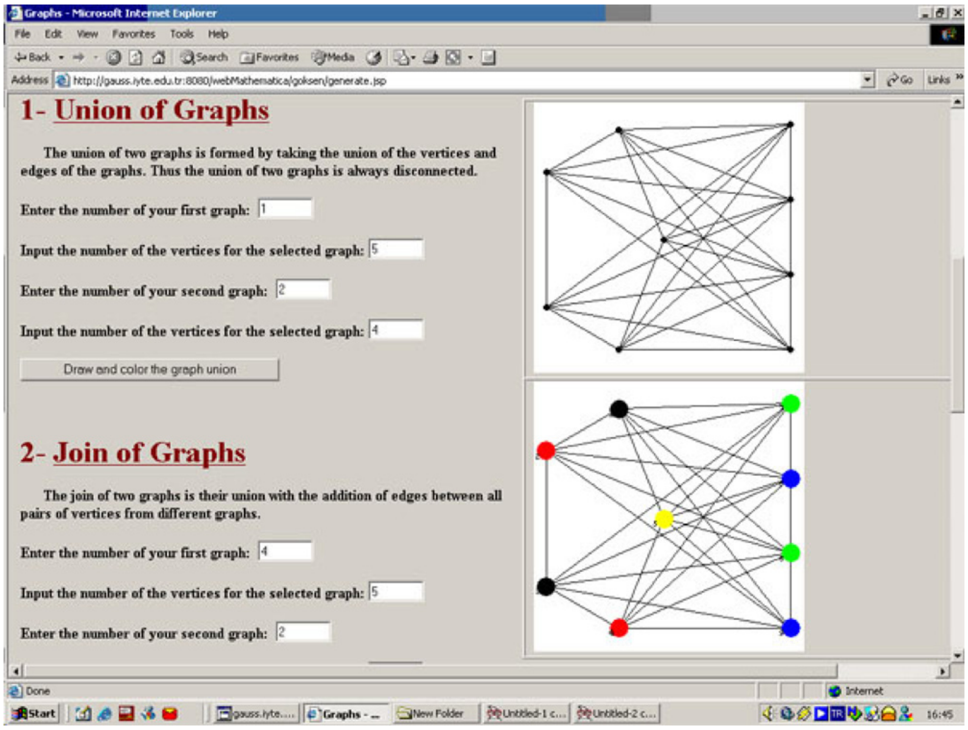
Grid graphs are bipartite, for the vertices can be partitioned like the squares on a chessboard. Their chromatic number is two.

If you enter the number of the vertices of rows and columns in a grid graph you will see the drawing of that graph which was colored properly with two colors.

5

3

Fig. 1. Vertex coloring of a grid graph with *webMathematica*.


Graphs - Microsoft Internet Explorer
http://gauss.iyte.edu.tr:8080/webMathematica/goksen/generate.jsp
Draw and color the graph union

1- Union of Graphs

The union of two graphs is formed by taking the union of the vertices and edges of the graphs. Thus the union of two graphs is always disconnected.

Enter the number of your first graph: 1

Input the number of the vertices for the selected graph: 5

Enter the number of your second graph: 2

Input the number of the vertices for the selected graph: 4

2- Join of Graphs

The join of two graphs is their union with the addition of edges between all pairs of vertices from different graphs.

Enter the number of your first graph: 4

Input the number of the vertices for the selected graph: 5

Enter the number of your second graph: 2

Fig. 2. Graph operations and coloring with *webMathematica*.

adjacent if and only if $P(x) \bigcap P(y) \neq \emptyset$. Then each proper vertex coloring of $G$ yields a schedule with the vertices in any color class representing the schedule on a particular day. Thus $\chi(G)$ gives the minimum number of days required for the bus schedule. The Mathematica commands for this solution are as follows:

```
<<DiscreteMath‘ColorG‘
k=Input["Input the number of locations that
will be visited by the tour bus companies"];
S=Table[Input["List the labels for the
tour bus companies that
will visit this location"],k];
```

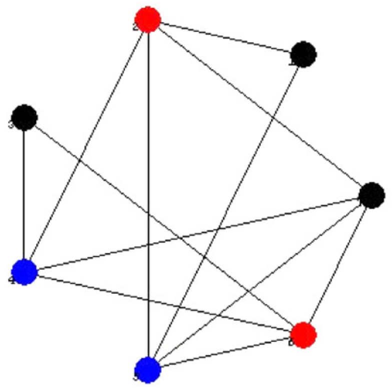


Fig. 3. A view of colored graph of the tour bus companies.

```
Union[Flatten[Table[KSubsets[S[[i]],2],
{i,k}],1]];
ColorVertices[t=DrawG[%]];
h=VertexColoring[t]; d=ChromaticNumber[t];
Print[d "days are required and you can
see below the label of the bus companies
in the same parenthesis which are on the
same day"]
Table[Flatten[Position[h,i],2],{i,Max[h]}]
```

Three days are required and you can see below the bus companies in which are on the same day in each set of parentheses.

{{1, 3, 7}, {2, 6}, {4, 5}}

### 2.2. Graphing of binomial trees

We used the Combinatorica package and developed a module to draw binomial trees with *Mathematica* (Fig. 4). The module DrawG must be added to the package DiscreteMath ‘Combinatorica’, and then the package must be loaded.

```
DrawG[elist_]:=
  Module[{el=elist,vl,size,nv},
   # el: list of edges, vl: list of vertices
     {size,vl}={Length[#],Union
   [Flatten[#]]} @ el;
     el=List[#]& /@ el;
     vl=CompleteGraph[Length[vl]][[2]];
     Graph[el,vl]
  ]
```

The following module BinomTree[n], draws the binomial tree with *a* vertices. BinomTree[a_]:=

```
  Module[{tt},
     tt=Flatten[
           Table[{2^n*(2m-1),2^(n+1)*m},{n,0,a-1},
             {m,1,2^(a-n-1)}],
        1];
     BinomialTree=DrawG[tt];
     ShowLabeledGraph[RootedEmbedding[%]];
     Print[‘‘BinomialTree"]
  ]
```

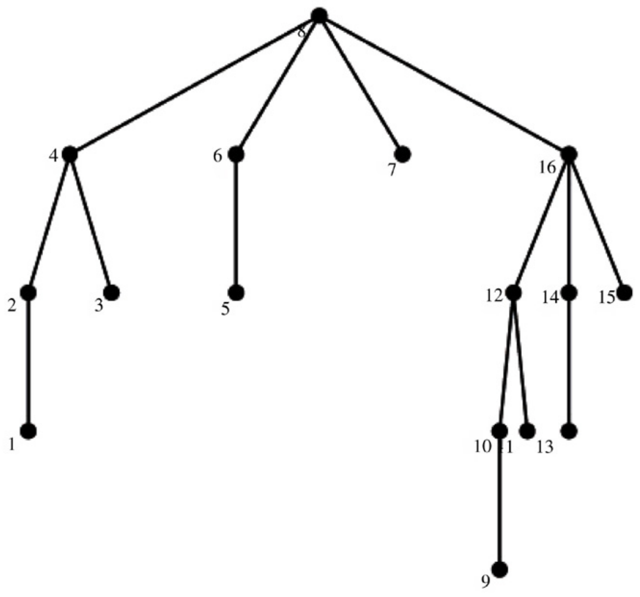


Fig. 4. A view of $B_4$ with *Mathematica*.

### 2.3. Tenacity of a graph

We developed a module that computes the tenacity of graphs: you can input the number of the vertices of a binomial tree and get its tenacity:

```
BinomTree[n]
bt=BinomialTree;
g=Edges[bt]; rr=Max[g]; d=Range[rr];
   b=Subsets[d];
t=Intersection[b,Complement[b,{{},d}]];
   lt=Length[t];
m=Table[ConnectedComponents[DeleteVertices
   [bt,t[[i]]]],{i,1,lt}];
m1=Table[Max[Table[r[i_]=Length[m[[i]][[f]]],
   {f,1,Length[m[[i]]]}]],
     {i,1,lt}];
m2=Table[Table[r[i_]=Length[m[[i]]],{i,1,lt}]];
s=Length[#]& /@ t;
Ten=(m1[[#]]+s[[#]])/m2[[#]]& /@ Range[lt];
Print["Tenacity=",Min[Ten]]
```

Instead of BinomTree[n], you can also use Cycle[n], CompleteGraph[n], Wheel[n], etc., and get the tenacity of these graphs.

## References

[1] K.I. Appel, W. Haken, J. Koch, Every planar map is four colorable I: Discharging, Illinois Journal Mathematics 21 (3) (1977) 429–490.
[2] G.D. Birkhoff, D.C. Lewis, Chromatic polinomials, Transactions American Mathematical Society 60 (1946) 355–451.
[3] H.L. Bodlaender, On the complexity of some coloring games, International Journal of Foundations of Computer Science 2 (2) (1991) 133–148.
[4] S.A. Choudum, N. Priya, Tenacity of complete graph products and grids, Networks 34 (3) (1999) 192–196.
[5] B.F. Cooper, M. Bawa, N. Daswani, S. Marti, H. García-Molina, Authenticity and availability in PIPE networks, Future Generation Computer Systems 21 (2005) 391–400.

[6] M. Cozzens, D. Moazzami, S. Stueckle, The tenacity of a graph, in: Proc Seventh International Conf. on the Theory and Applications of Graphs, Wiley, New York, 1995, pp. 1111–1122.
[7] C. Curti, T. Ferrari, L. Gommans, S. Ourdenaarde, E. Ronchieri, F. Giacomini, C. Vistoli, On advance reservation of heterogeneous network paths, Future Generation Computer Systems 21 (2005) 525–538.
[8] D. Gao, Y. Kinouchi, K. Ito, X. Zhao, Neural networks for event extraction from time series: A back propagation algorithm approach, Future Generation Computer Systems 21 (2005) 1096–1105.
[9] D. Ghosal, B.K. Poon, K. Kong, P2P contracts: A framework for resource and service exchange, Future Generation Computer Systems 21 (2005) 333–347.
[10] G. Echevarría, A. Iglesias, A. Gálvez, Extending neural networks for B-spline surface reconstruction, Lecture Notes in Computer Science 2330 (2002) 305–314.
[11] A. Iglesias, G. Echevarría, A. Gálvez, Functional networks for B-spline surface reconstruction, Future Generation Computer Systems 20 (2004) 1337–1353.
[12] G. Jonathan, Y. Jay, Graph Theory and Its Applications, CRC Press, 1999.
[13] M.A. Klopotek, Very large Bayesian multinets for text classification, Future Generation Computer Systems 21 (2005) 1068–1082.
[14] Z. Liu, A. Liu, C. Wang, Z. Niu, Evolving neural network using real coded genetic algorithm (GA) for multispectral image classification, Future Generation Computer Systems 20 (2004) 1119–1129.
[15] H. Martin, Looking for narrow interfaces in automatic differentiation using graph drawing, Future Generation Computer Systems 21 (2005) 1418–1425.
[16] H.A.D. Nascimento, P. Eades, User hints: A frameworrk for interactive optimization, Future Generation Computer Systems 21 (2005) 1177–1191.
[17] R. Neruda, P. Kudova, Learning methods for radial basis function networks, Future Generation Computer Systems 21 (2005) 1131–1142.
[18] S. Ourdenaarde, Z. Hendrikse, F. Dijkstra, L. Gommans, C. Laat, R.J. Meijer, Dynamic paths in multi-domain optical networks for grids, Future Generation Computer Systems 21 (2005) 539–548.
[19] S. Pemmaraju, S. Skiena, Computational Discrete Mathematics-Combinatorics and Graph Theory with *Mathematica*, Cambridge University Press, 2003.
[20] S. Skiena, Implementing Discrete Mathematics-Combinatorics and Graph Theory with *Mathematica*, Addison-Wesley Publishing Company, 1990.
[21] T.L. Soaty, P.C. Kainen, The Four Color Problem, Dover, New York, 1986.
[22] K. Thulasiraman, M.N.S. Swamy, Graphs: Theory and Algorithms, John Wiley and Sons, 1992.
[23] Ü. Ufuktepe, G. Bacak, T. Beseri, Graph Coloring with webMathematica, in: Lecture Notes in Computer Science, vol. 3039, Springer-Verlag, 2004, pp. 376–381.
[24] T. Wickham-Jones, *webMathematica* A User Guide, Wolfram Research, Inc., 2002.
[25] S. Wolfram, The *Mathematica* Book, Cambrigde Univ. Press, 1996.

**Ünal Ufuktepe** was born in Kayseri-Develi. He earned a B.S. degree in Mathematics from the Middle East Technical University in Ankara, Turkey in 1983, an M.Sc. degree from Ankara University in 1986, his Mathematics Education Certificate in 1990 at Gazi University, and his Ph.D. in Mathematics from the University of Missouri-Columbia, USA in 1996. He worked at different universities, colleges and high schools as a TA, Instructor and teacher. He worked at Izmir Institute of Technology as the founder of department of mathematics and vice-dean of the Science Faculty. He is currently working at the same institute as a full time associate professor. Dr. Ufuktepe has published numerous articles and books on Mathematics related topics, including probabilistic analysis, Mathematica applications on time scales and graph coloring, dynamical systems and math anxiety.

**Goksen Bacak** is a lecturer in Yasar University, Turkey. She earned her B.S. degree in Mathematics at the Educational Faculty of Marmara University, Turkey in 1999 and her M.Sc. Degree in Mathematics from Izmir Institute of Technology in 2004. She is currently a Ph.D. candidate in the department of Mathematics in Ege University, Turkey. Her research interest is Graph Theory, especially vulnerability in graphs.