Int. J. Advance Soft Compu. Appl, Vol. 8, No. 2, July 2016
ISSN 2074-8523

d-Edge Sum Labeling of Graphs

D. Ahima Emilet[1] and Indra Rajasingh[1]

[1]VIT University,
Vandalur- Kelambakkam Road, Chennai, India.
e-mail: ahi.180492@gmail.com, indrarajasingh@yahoo.com

Abstract

In this paper we introduce a new labeling of graphs called d-edge sum labeling and study the same as a vertex coloring problem. Let $l : E(G) \rightarrow N$ be a labeling of the edges of a graph G by positive integers. Define $c(u) = \sum_{(u,v)\in E(G)} l(u,v) + d(u)$, where $d(u)$ denotes the degree of u. We call l a d-edge sum labeling if $c(u) \neq c(v)$, for every pair of adjacent vertices u and v in G. The d-edge sum number of a graph G, denoted by $\eta_{ds}(G)$, is the least positive k such that G has a d-edge sum labeling with $\{1,2,\ldots,k\}$ as the set of labels. In this paper we obtain $\eta_{ds}(G)$ for some special classes of graphs.

Keywords: *Butterfly network, d-edge sum, Edge-labeling, Hhypertree, slim tree.*

1 Introduction

Graph coloring is one of the most studied subjects in graph theory. In 2004, Karonski et al. [6] have stated that a weighting of the edges of a graph with integer weights gives rise to a weighting of the vertices, the weight of a vertex being the sum of the weights of its incident edges. The number of consecutive integer weights needed so that all vertices receive different weights has been called the irregularity strength of a graph. Weighting in other words denotes the label of each edge. Karonski, Luczak and Thomason [6] initiated the study of proper labeling. A proper labeling of a graph is an assignment of integers to some elements of the graph, which may be the vertices, the edges, or both of them, such that we obtain a vertex coloring via the labeling, such that adjacent vertices do not receive the same color, usually addressed as proper vertex coloring. Graph coloring is used in various research areas of computer science such as data mining, image segmentation, clustering, image capturing, networking and applications such as guarding an art gallery, physical layout segmentation, scheduling [3], biprocessor tasks, frequency assignment, map coloring, GSM mobile phone networks, exam timetabling and student timetabling [8]. In this paper we introduce a new labeling called d-edge sum labeling and compute d-edge sum number of complete bipartite graphs and certain networks such as wheels, butterfly and hypertrees.

2 *d*-Edge sum labeling problem

Karonski et al. introduced edge-labeling by sum [6].

Definition 2.1 [6] *We call a mapping* $f : E(G) \to \{1,2,\ldots,k\}$, *an edge-labeling by sum if for all* $u \in V(G)$, $c(u) = \sum_{(u,v)\in E(G)} f(u,v)$ *is a proper vertex coloring of* G.

In this paper we introduce a new labeling which is equivalent to edge-labeling by sum for regular graphs.

Definition 2.2 *Let* $l : E(G) \to \{1,2,\ldots,k\}$ *be a labeling of the edges of a graph* G *by positive integers. Define* $c(u) = \sum_{(u,v)\in E(G)} l(u,v) + d(u)$, *where* $d(u)$ *denotes the degree of* u. *We define the labeling* l *as* d*-edge sum labeling if* $c(u) \neq c(v)$, *for every pair of adjacent vertices* u *and* v *in* G. *The* d*-edge sum number of a graph* G, *denoted by* $\eta_{ds}(G)$, *is the least positive integer* k *such that* G *has a* d*-edge sum labeling with* $\{1,2,\ldots,k\}$ *as the set of labels.*

We observe that not every d-edge sum labeling is edge-labeling by sum. Fig. $1(a)$ illustrates d-edge sum labeling of a graph G. However this labeling is not an edge-labeling by sum; see Fig. $1(b)$. Conversely, not every edge-labeling by sum induces d-edge sum labeling. Fig. $1(c)$ illustrates edge-labeling by sum of a graph H. However this labeling is not a d-edge sum labeling; see Fig. $1(d)$.

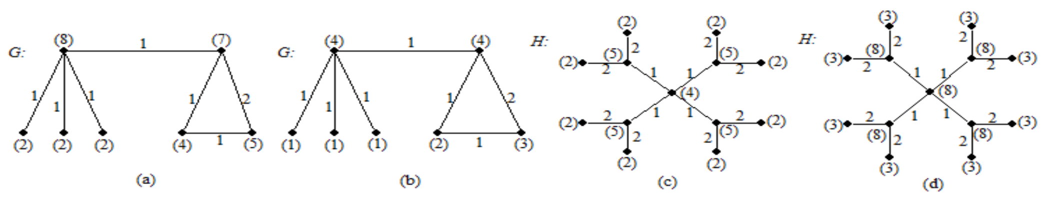

Fig. 1: The number within the paranthesis represents $c(u)$ (a) d-edge sum labeling of a graph G (b) Edge-labeling by sum of a graph G (c) Edge-labeling by sum of a graph H (d) d-edge sum labeling of a graph H.

The following result is an easy observation.

Theorem 2.3 *For a graph* G, $\eta_{ds}(G) = 1$ *if and only if no two adjacent vertices in* G *have the same degree.*

3 Some classes of graphs

3.1 Complete bipartite graph

Definition 3.1.1 *A graph G is bipartite with bipartition (X,Y) if $V(G)=X\cup Y$, $X\cap Y=\phi$ and every edge in G has one end in X and the other end in Y. G is a complete bipartite graph if every vertex in X is joined to every vertex in Y and is denoted by $K_{m,n}$ if $|X|=m$ and $|Y|=n$.*

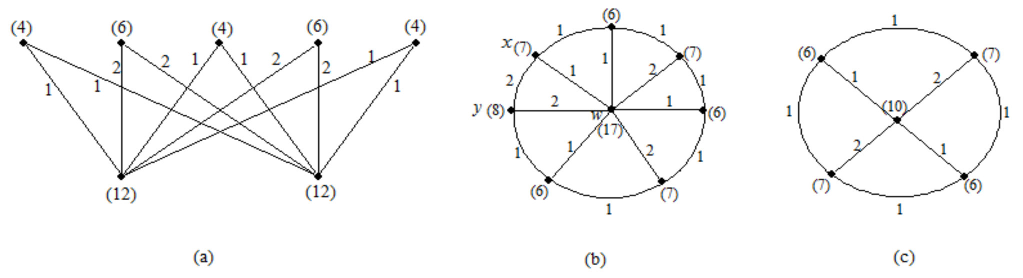

Fig. 2: (a) d-edge sum labeling of $K_{5,2}$ (b) d-edge sum labeling of W_8 (c) d-edge sum labeling of W_5

3.1.2 Algorithm d-edge sum $K_{m,n}$

Input : A complete bipartite graph $K_{m,n}$ with bipartition (X,Y), $|X|=m$, $|Y|=n$, $m\geq n\geq 2$, $6n\neq 5m$ when m is even and $6n\neq 5m-1$ when m is odd.

Algorithm :

Step 1. Order the vertices of X from left to right as $1,2,\ldots,m$.

Step 2. Label the edges incident on i^{th} vertex of X, i odd, as 1.

Step 3. Label the edges incident on i^{th} vertex of X, i even, as 2.

Output : $\eta_{ds}(K_{m,n})=2$

Proof of correctness : Let $e=(u,v)\in E(K_{m,n})$ with u in X and v in Y. All edges incident at u are labeled either 1 or 2. Hence $c(u)=2n$ or $3n$, according as u is an odd labeled vertex or an even labeled vertex of X respectively. On the other hand, when m is even, $c(v)=(\frac{m}{2}\times 1)+(\frac{m}{2}\times 2)+m=\frac{5m}{2}$ and when m is odd, $c(v)=\frac{(m-1)}{2}\times 1+\frac{(m-1)}{2}\times 2+m+1=\frac{5m-1}{2}$. When $c(u)=2n$, if

$c(v)=\frac{5m}{2}$, then $c(u)=c(v)$ implies $4n=5m\geq 5n$, a contradiction. If $c(v)=\frac{5m-1}{2}$, then $c(u)=c(v)$ implies $4n=5m-1\geq 5n-1$, a contradiction. Therefore in either case, $c(u)\neq c(v)$. On the other hand, when $c(u)=3n$, $c(u)\neq c(v)$ only when $6n\neq 5m$, m even and $6m\neq 5m-1$, m odd. See Fig. $2(a)$.

Theorem 3.1.3 *Let $K_{m,n}$ be the complete bipartite graph with $m\geq n\geq 2$, $6n\neq 5m$ for m even and $6n\neq 5m-1$ for m odd. Then $\eta_{ds}(K_{m,n})=2$.*

3.2 Wheel graph

Definition 3.2.1 *A wheel graph denoted by W_n is a graph with n vertices $n\geq 4$ and $2(n-1)$ edges, formed by connecting a single vertex called centre vertex to all vertices of an $(n-1)$-cycle.*

3.2.2 Algorithm d-edge sum W_n

Input: Wheel graph W_n, $n\geq 5$

Algorithm:

Step 1. For n odd, $n\geq 5$, label the cycle edges as 1 and the edges incident at the central vertex alternately as 1 and 2.

Step 2. For n even, $n\geq 6$, label all the cycle edges except one edge as 1, and the unlabeled edge say (x,y) as 2. Label the edges incident at the central vertex w alternately as 1 and 2 beginning with labeling edges (x,w) as 1 and (y,w) as 2.

Output: $\eta_{ds}(W_n)=2$ for $n\geq 5$.

Proof of correctness: Let W_n be a wheel graph with n vertices, where $n\geq 5$.

Case 1: n is even, $n\geq 6$: Let $e=(u,v)\in E(W_n)$. If u is the centre vertex then $c(u)=(n-2)+2(n-2)+1=3n-5$ and $c(v)=6,7$ or 8. Therefore $c(u)\neq c(v)$. If (u,v) is a cycle edge and $(u,v)=(x,y)$, then $c(u)=7$ and $c(v)=8$, whereas all the other values of $c(u)$ are alternating 6 and 7 on the cycle. Thus $c(u)\neq c(v)$, $\forall\ (u,v)\in E(W_n)$. See Fig. $2(b)$.

Case 2: n is odd, $n\geq 5$: Let $e=(u,v)\in E(W_n)$. If u is the centre vertex and v is a cycle vertex whose edge is labeled 1 or 2,

then $c(u)=\left(\frac{n-1}{2}\right)+2\left(\frac{n-1}{2}\right)+(n-1)=\frac{5(n-1)}{2}$ and $c(v)=6$ or 7.

Hence $c(u) \neq c(v)$, $\forall$ $(u,v) \in E(W_n)$. Further if $e = (u,v)$ is an edge where both u and v are cycle vertices, then $c(u) = 6$ and $c(v) = 7$. See Fig. $2(c)$.

Theorem 3.2.3 *The wheel graph* W_n *admits* d *-edge sum labeling with* $\eta_{ds}(W_n) = 2$, *for* $n \geq 5$.

3.3 Butterfly networks

A most popular bounded - degree derivative network of the hypercubes is called a butterfly network.

Definition 3.3.1 *[7] The* r *- dimensional butterfly network, denoted by* $BF(r)$, *has a vertex set* $V = \{(x;i) : x \in V(Q_r), 0 \leq i \leq r\}$. *Two vertices* $(x;i)$ *and* $(y;j)$ *are linked by an edge in* $BF(r)$ *if and only if* $j = i+1$ *and either*

(i) $x = y$, *or*

(ii) x *differs from* y *in precisely the* j^{th} *bit.*

For $x = y$, *the edge is said to be a straight edge. Otherwise, the edge is a cross edge. For fixed* i, *the vertex* $(x;i)$ *is a vertex on level* i.

Remark 3.3.2 *All edges in* $BF(r)$ *join a vertex in level* i *with a vertex in level* $i+1$, $0 \leq i \leq r-1$.

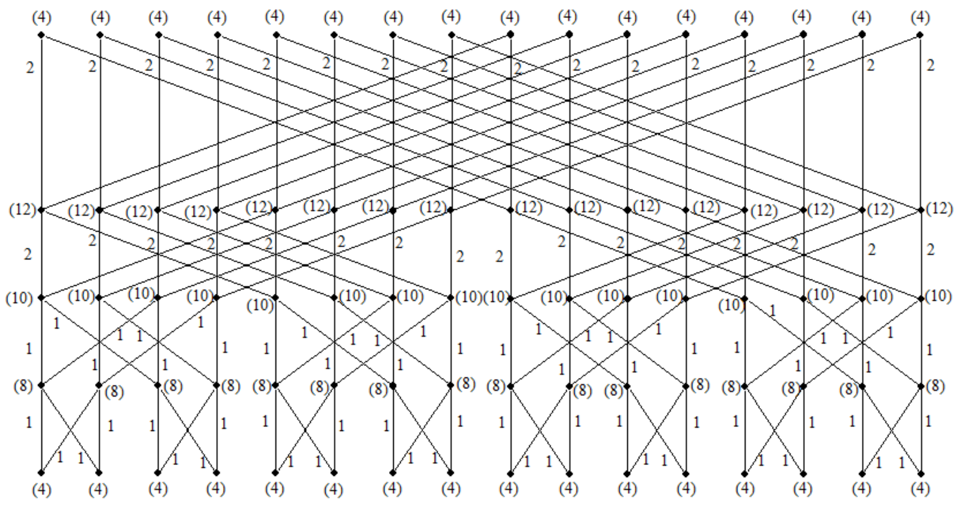

Fig. 3: d -edge sum labeling of BF(4)

3.3.3 Algorithm d -edge sum $BF(r)$

Input : r - dimensional butterfly network $BF(r)$, $r \geq 1$

Algorithm :

Label the edges of $BF(r)$ as follows:

Step 1. When r is even, label the straight edges and cross edges between level i and $i+1$ and those between $i+1$ and $i+2$ as 1 whenever $i \equiv 0 \, mod \, 4$ and as 2 whenever $i \equiv 2 \, mod \, 4$, $0 \le i \le r$.

Step 2. When r is odd, repeat the same labeling for $0 \le i \le r-3$ and label all the edges between levels $(r-1)$ and r as 1 whenever $r \equiv 1 \, mod \, 4$ and as 2 whenever $r \equiv 3 \, mod \, 4$.

Output : $\eta_{ds}(BF(r)) = 2$ for $r \ge 1$.

Proof of correctness : The r-dimensional butterfly network $BF(r)$, $r \ge 1$, is a biregular graph with vertices of degree 2 at levels 0 and $r+1$ and all other vertices of degree 4. The algorithm d-edge sum $BF(r)$ ensures that if each i^{th} level vertex $i \ne 0, r+1$, has two edges labeled as 1 and two edges labeled as 2 incident at it, then each $i+1^{th}$ level vertex $i \ne 0, r+1$ has all four edges incident at it labeled 1 or all four edges labeled 2. Maximum value of $c(u)$, when u is in level 0 or $r+1$ is 6. Minimum value of $c(v)$, when v is in level 1 or r is 8. This implies $c(u) \ne c(v)$, $\forall (u,v) \in E(G)$. See Fig. 3.

Theorem 3.3.4 *The r-dimensional butterfly network $BF(r)$ admits d-edge sum labeling and $\eta_{ds}(BF(r)) = 2$.*

3.4 Benes network

Definition 3.4.1 [7] *The r-dimensional benes network denoted by $BB(r)$ consists of back-to-back butterfly networks. The $BB(r)$ has $2r+1$ levels, each with 2^r vertices. The middle level in $BB(r)$ is shared by these butterfly networks.*

Applying algorithm d-edge sum $BF(r)$ for levels 0 to $2r+1$ of $BB(r)$ we arrive at the following result.

Theorem 3.4.2 *The r-dimensional benes network $BB(r)$ admits d-edge sum labeling and $\eta_{ds}(BB(r)) = 2$.*

4 Some classes of tree derived networks

A tree is a connected acyclic graph. Trees are the most fundamental graph theoretic models used in many fields such as information theory, operations research, theory of electrical and design networks.

4.1 Hypertree

A hypertree is an interconnection topology for incrementally expansible multicomputer systems, which combines the easy expansibility of tree structures with the compactness of the hypercube; that is, it combines the best features of the binary tree and the hypercube. These two properties make this topology particulary attractive for implementation of multiprocessor networks of the future, where a complete computer with a substantial amount of memory can fit on a single VLSI chip [4].

The basic skeleton of a hypertree is a complete binary tree T_r. We denote an r-level hypertree as $HT(r)$. The root vertex is said to be at level 0. For any vertex v at level i, the edge connecting it to the vertex in level $i-1$ is addressed as parent edge and the two edges connecting it to its left and right children are called left edge and right edge incident at v, $1 \le i \le r$. Every vertex v in any level i, $i \ne 0$ is adjacent to its corresponding vertex at the same level and such edges are addressed as horizontal edges. $HT(r)$ has $2^{r+1}-1$ vertices and $3(2^r-1)$ edges.

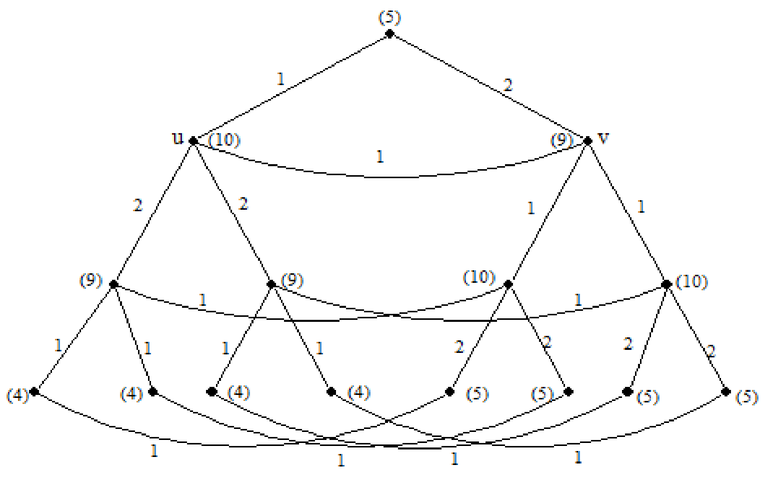

Fig. 4: d-edge sum labeling of Hypertree HT(3)

4.1.1 Algorithm d-edge sum $HT(r)$

Input: Hypertree $HT(r)$, $r \ge 1$

Algorithm:

Step 1. Label the left edge and right edge incident at the root vertex as 1 and 2 respectively.

Step 2. Let u and v be the left and right child of the root vertex. Label the edges at alternate levels of the complete binary subtree rooted at u as 1 and 2 beginning with 1. Similarly, label the edges at alternate levels of the complete binary tree rooted at v as 2 and 1 beginning with 2.

Step 3. Label the horizantal edges as 1.

Output : $\eta_{ds}(HT(r)) = 2$

Proof of correctness : Let $e = (u,v) \in E(HT(r))$. Every vertex u other than the root vertex has one horizontal edge incident at it and it is labeled 1. Again, every vertex v at level i, $i \neq 0, r$, has a parent edge, a left edge, a right edge and a horizontal edge incident at it. By labeling algorithm, if parent edge is labeled 1, left and right edges are labeled 2 and vice versa. Hence if u and v are adjacent vertices of degree 4 at different levels then $c(u) = \sum_{(u,v)\in E(G)} l(u,v) + d(u) = 9$ and $c(v) = \sum_{(v,u)\in E(G)} l(v,u) + d(v) = 10$. If u and v are adjacent vertices of degree 4 at the same level, then if u is in the left binary tree then v is the corresponding vertex in the right binary tree then $c(u) = 10$ and $c(v) = 9$. If u is the root vertex, then $c(u) = 5$ which is less than $c(v)$ if v is the right or left child of v. If u and v are adjacent vertices of degree 2 at level r, then $c(u) = 5$ and $c(v) = 4$. See Fig. 4.

Theorem 4.1.2 *The hypertree* $HT(n)$ *admits* d *-edge sum labeling and* $\eta_{ds}(HT(r)) = 2$, $n \geq 2$.

4.2 Slim tree

Definition 4.2.1 [2] *The* n^{th} *slim tree* $ST(n)$, $n \geq 2$, *denoted by* $ST(n) = (V, E, u, l, r)$, *where* V *is the node set,* E *is the edge set and* u , l , r *are vertices addressed as root node, left node and right node respectively is recursively defined as follows:*

1. $ST(2)$ *is the complete graph* K_3 *with its nodes labeled as* u , l *and* r .

2. *The* s^{th} *slim tree* $ST(s)$, *with* $s \geq 3$ *is composed of a root node* u *and two disjoint copies of* $(s-1)^{th}$ *slim trees as the left subtree and right subtree, denoted by* $ST^{l}_{(n-1)} = (V_1, E_1, u_1, l_1, r_1)$ *and* $ST^{r}_{(n-1)} = (V_2, E_2, U_2, l_2, r_2)$, *respectively. To be specific,* $ST(n) = (V, E, u, l, r)$ *is given by* $V = V_1 \cup V_2 \cup u$, $E = E_1 \cup E_2 \cup (u,u_1),(u,u_2),(r_1,l_2), l = l_1, r = r_2$.

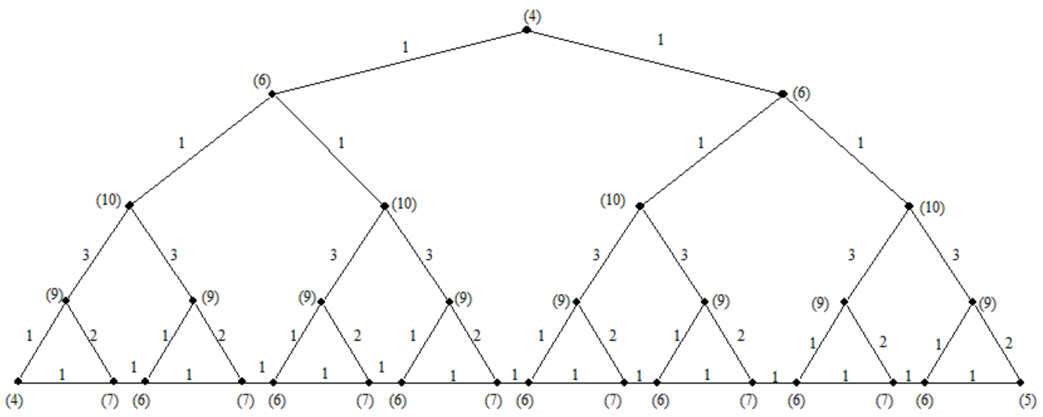

Fig. 5: d -edge sum labeling of Slim tree, ST(5)

4.2.2 Algorithm d -edge sum $ST(n)$

Input : Slim tree $ST(n)$, $n \geq 3$

Algorithm :

Let the root vertex be at level 0. Vertices at distance i from the root vertex are said to be at level i.

Step 1. Label all the horizontal edges as 1.

Step 2. Label the left edges incident on vertices in level $n-1$ as 1 and the right edges incident on vertices in level $n-2$ as 2.

Step 3. Label the left and right edges incident on vertices at level $n-2$ as 3.

Step 4. Label the edges incident at vertices in levels $n-2$ to 0 as follows:

(i) Label the edges between levels $n-4k+2$ and $n-4k+1$ and edges between levels $n-4k+1$ and $n-4k$, $k=\{1,2,...\}$ as 1.

(ii) Label the edges between levels $n-4k$ and $n-4k-1$ and edges between levels $n-4k-1$ and $n-4k-2$, $k=\{1,2,...\}$ as 2.

Output : $\eta_{ds}(ST(n)) \leq 3$

Proof of correctness : Let $e=(u,v) \in E(ST(n))$. The sum $c(u)$, for all u in the same level are all equal. For every vertex at level $n-1$, $c(u)$ is 9 and for any vertex at level n, $c(u)$ is less than or equal to 9. Similarly, $c(u)$ of vertices at level $n-2$ is 10. Further $c(u)$ of vertices at levels beginning from $n-3$ to 1 follow the pattern $(6,7,9,8)^i$ followed by $\phi,6$ or $6,7$ or $6,7,9$ according as $n \equiv 0,1,2,3 \, mod \, 4$ respectively, $i \geq 0$. Hence $c(u) \neq c(v)$.

Theorem 4.2.3 *The slim tree $ST(n)$ admits d -edge sum labeling and $\eta_{ds}(ST(n)) \le 3$, for $n \ge 3$.*

5 Conclusion

A new graph labeling called d -edge sum labeling is defined. Finally, d -edge sum labeling of complete bipartite graphs and some classes of graphs such as cycle, wheel graphs, butterfly network, benes network, slim tree and hypertree are investigated.

ACKNOWLEDGEMENTS
This work is supported by Project No. SR/S4/MS: 846/13, Department of Science and Technology, SERB, Government of India. The authors would like to thank the anonymous referees for their comments and suggestions. These comments and suggestions were very helpful for improving the quality of the paper.

References

[1] Dudek, A., Wajc, D. 2011. On the complexity of vertex-coloring edge-weightings. *Discrete Mathematics and Theoretical Computer Science*. Vol 13, No. 3, 45-50.

[2] Hung, C.,-N., Hsu, L.,-H., and Sung, T., -Y. 1999. Christmas tree: a versatile 1-fault-tolerant design for token rings. *Information Processing Letters.* Vol 72, No.1- 2, 55- 63.

[3] Marx, D. 2004. Graph coloring problems and their applications in scheduling. *Periodica Polytechnica. Mechanical Engineering*. Vol 48, No. 1, 11-16.

[4] Goodman, J., R. and Sequin, C., H. 1981. A multiprocessor interconnection topology. *IEEE Transactions on Computers.* Vol -c-30, No. 12, 923- 933.

[5] Kalkowski, M., Karonski, M., L., Pfender, F. Vertex coloring edge weightings with integer weights at most 6 . *http://www.math.uni-rostock.de/ pfender/papers/23.pdf*

[6] Karonski, M., Luczak, T., Thomason, A. 2004. Edge weights and vertex colours. *Journal of Combinatorial Theory. Series B.* Vol 91, No. 1, 151-157.

[7] Xu, J. Topological structure and analysis of interconnection Networks. *Kluwer Academic Publishers.* Boston.

[8] Shrinivas, S.,G., Vetrivel, S. and Elango, N.,M. 2010. Applications of graph theory in computer science an overview. *International Journal of Engineering Science and Technology.* Vol 2, No. 9, 4610-4621.