Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 57 (2015) 716 – 726

Procedia Computer Science

3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015)

Tree Derived Architectures with Decycling Number equal to Cycle Packing Number

Paul D[a,] *, Indra Rajasingh[a], R Sundara Rajan[a]

[a] School of Advanced Sciences, VIT University, Chennai-600127, India.

Abstract

The decycling number of a graph G, denoted by $\nabla(G)$, is the smallest number of vertices that can be deleted from G so that the resultant graph contains no cycles. The cycle packing number of a graph G denoted by $c(G)$, is the maximum number of vertex disjoint cycles in G. It is clear that $c(G) \leq \nabla(G)$. We find certain networks for which decycling number equals the cycle packing number.

Keywords: Decycling number; cycle packing number; vertex-disjoint cycles; hypertree; slim tree; Christmas tree; k -rooted sibiling tree; X -tree; l -siblings tree; l -complete binary tree; k -dimensional generalised binary fat tree;

1 Introduction

Let $G(V, E)$ be a simple graph with vertex set $V(G)$ and edge set $E(G)$. A set of vertices of G whose removal leaves an acyclic graph is referred to as a decycling set or a feedback vertex set of G. The minimum cardinality of the decycling sets in G is defined to be the decycling number and is denoted by $\nabla(G)$. Determining the decycling number of a graph is equivalent to finding the greatest order of an induced forest in G. The problem of determining the decycling number $\nabla(G)$ of a network G is NP-complete even for planar graphs, bipartite graphs and perfect graphs[1]. A minimum decycling set in a Wavelength Division Multiplexing Network[2], sets up routes between given pairs of nodes in the network and determines the minimum number of wavelength converters that are used in order to reduce the number of wavelengths used to set up the communications. Decycling sets also have applications in combinotorial circuit designs[3], deadlock prevention in operating systems[4,5], monopolies in synchronous distributed systems[6,7], constraint satisfaction problem and bayesian inference in artificial intelligence[8] and VLSI chip design[9].

* Corresponding author. Tel.:+91-948-630-4300;.
E-mail address:paul.d.joe@gmail.com.

1877-0509 © 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Peer-review under responsibility of organizing committee of the 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015)
doi:10.1016/j.procs.2015.07.460

Another parameter that is closely related to the decycling number is the cycle packing number, which is the maximum number of vertex-disjoint cycles in a graph G. We denote this parameter by $c(G)$. Determining the cycle packing number of a graph is also known to be NP-complete[10]. It has applications in computational biology and reconstruction of evolutionary trees[11]. It also has application in kidney exchange programs[12].

2 Decycling Number of Certain Graphs

Chang et al.[13] have observed the following result.

Lemma 2.1 *For any graph* G, $\nabla(G) \geq c(G)$, *where* $\nabla(G)$ *is the decycling number of* G *and* $c(G)$ *is the cycle packing number of* G.

In this paper, we compute the decycling number $\nabla(G)$ of G, when G is a hypertree, slim tree, Christmas tree, k-rooted sibiling tree, X-tree, l-siblings tree, l-complete binary tree or a fat tree. Our stratergy is to obtain $c(G)$ for a given graph G and prove that the lower bound in lemma 2.1 is sharp for the graph.

2.1 Hypertree

A tree is a connected graph that contains no cycles. The most common type of tree is the binary tree. It is so named because each node can have at most two descendents. A binary tree is said to be a complete binary tree if each internal node has exactly two descendents. These descendents are described as left and right children of the parent node. Binary trees are widely used in data structures because they are easily stored, easily manipulated, and easily retrieved. Also, many operations such as searching and storing can be easily performed on tree data structures. Furthermore, binary trees appear in communication pattern of divide-and-conquer type algorithms, functional and logic programming, and graph algorithms. A rooted tree represents a data structure with a hierarchical relationship among its various elements[14].

For any non-negative integer r, the complete binary tree of height $r-1$, denoted by T_r, is the binary tree where each internal vertex has exactly two children and all the leaves are at the same level. Clearly, a complete binary tree T_r has r levels and level $i, 0 \leq i \leq r-1$, contains 2^i vertices. Thus, T_r has exactly $2^r - 1$ vertices.

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[15].

The basic skeleton of a hypertree is a complete binary tree T_n. Here the nodes of the tree are labeled as follows: The root node receives label 1. The root is said to be at level 0. Labels of left and right children are formed by appending a 0 and 1, respectively to the labels of the parent node. Here the children of the nodes x are labeled as $2x$ and $2x+1$. Additional links in a hypertree are horizontal and two nodes are joined in the same level i of the tree if their label difference is 2^{i-1}. We denote an n-level hypertree as $HT(n)$. It has $2^{n+1}-1$ vertices and $3(2^n-1)$ edges.

Rajasingh et al.[16] have proved that Figure 1(a) and Figure 1(b) are isomorphic.

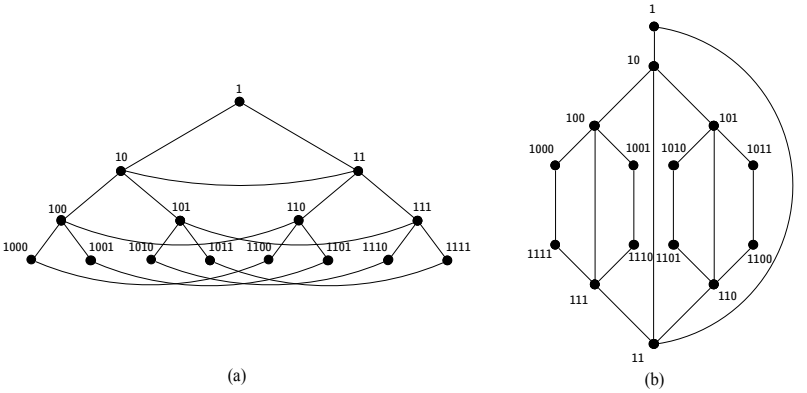

Figure 1: (a) Original drawing of $HT(3)$ (b) Drawing proposed by Rajasingh et al.

Definition 2.2 *A theta graph $\theta(a,b,c)$ consists of a pair of vertices u, v with three internally disjoint paths of lengths a, b and c joining u to v, where a, b, c are positive integers. The vertices u and v are called the poles of $\theta(a,b,c)$.*

The theta graph $\theta(1,3,3)$ is a subgraph of the hypertree $HT(n)$ and refer to it simply as θ-graph. It has 6 vertices and 7 edges.

Remark 2.3 *The hypertree $HT(n)$, $n \geq 3$, contains 2^{n-2} number of vertex disjoint θ-graphs.*

Lemma 2.4 *Let G be a θ-graph. Then $c(G) = 1$.*

Proof. Any cycle in G has to pass through the poles of G. Hence $c(G) = 1$.

Lemma 2.5 *Let G be the Hypertree $HT(n)$, $n \geq 1$. Then* $\nabla(G) \geq \begin{cases} \frac{1}{3}(2^n - 1) & \text{if } n \equiv 0 (mod\ 2) \\ \frac{1}{3}(2^n + 1) & \text{if } n \equiv 1 (mod\ 2) \end{cases}$.

Proof. We prove the result by induction on n.

Case (i): $n \equiv 0 (mod\ 2)$

All the three cycles in $HT(2)$ share a common vertex. Hence $c(G) = 1$ proving the result when $n = 2$. Assume the result to be true for $n = 2k$, $k \geq 1$. Consider $HT(2k+2)$. By Remark 2.3, there are 2^{2k} vertex disjoint θ-graphs in $HT(2k+2)$. Deletion of the vertices of these θ-graphs results in $HT(2k)$

Therefore by Lemma 2.4, $c(HT(2k+2)) \geq c(HT(2k)) + 2^{2k} = \frac{1}{3}(2^{2k-1}) + 2^{2k} = \frac{1}{3}(2^{2k+2} - 1)$.

Case (ii): $n \equiv 1 (mod\ 2)$

When $n = 1$, the result is trivial. Assume the result to be true for $n = 2k-1$, $k \geq 1$. Consider $HT(2k+1)$. By Remark 2.3, there are 2^{2k-1} vertex disjoint copies of θ-graphs in $HT(2k+1)$. Deletion

of the vertices of these θ -graphs results in $HT(2k-1)$ .Therefore by Lemma 2.4, $c(HT(2k+1)) \geq c(HT(2k-1)) + 2^{2k-1} = \frac{1}{3}(2^{2k-1}+1) + 2^{2k-1} = \frac{1}{3}(2^{2k+1}+1)$.

In both cases, the result follows from Lemma 2.1.

2.1.1 Algorithm Decycling Number of Hypertree $HT(n)$

Input*:* Hypertree $HT(n)$, $n \geq 3$

Algorithm:

(i) For n even, choose all the vertices in level $n-1, n-3, \ldots, 3, 1$.

(ii) For n odd, choose all the vertices in level $\{n-1, n-3, \ldots, n-(n-2)\} \cup \{1\}$.

Output: $\nabla(HT(n)) = \begin{cases} \frac{1}{3}(2^n - 1), n \ \textit{even} \\ \frac{1}{3}(2^n + 1), n \ \textit{odd} \end{cases}$

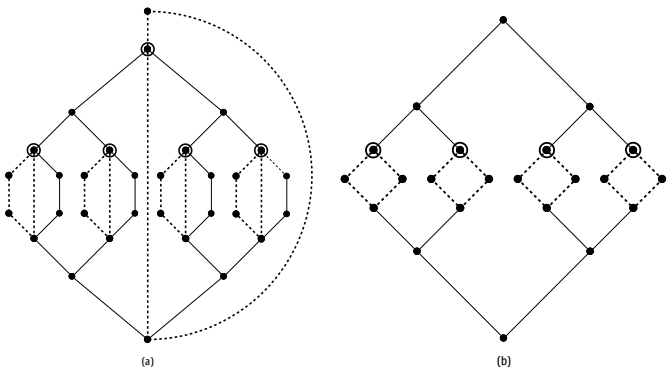

Figure 2: (a) darken cycles and circled vertices constitute cycle packing and minimum decycling set of *HT*(4)
(b) darken cycles and circled vertices constitute cycle packing and minimum decycling set of *l*-T_3

Proof of Correctness:

Let C be any cycle in $HT(n)$, $n \geq 2$. Then C contains at least one vertex in each level i of $HT(n)$, where $i \in \{n-1, n-3, \ldots, n-(n-2)\} \cup \{1\}$ for n odd and $i \in \{n-1, n-3, \ldots, 3, 1\}$ for n even. Let S be the set of all vertices in levels $n-1, n-3, \ldots, 3, 1$, when n is even and the set of all vertices in level $n-1, n-3, \ldots, 2$ together with vertex in level 1, when n is odd. Removal of vertices in S ensures that it destroys all the cycles in $HT(n)$.

Theorem 2.6 *Let* G *be the Hypertree* $HT(n)$, $n \geq 1$. *Then* $\nabla(G) = \begin{cases} \frac{1}{3}(2^n - 1) & \textit{if } n \equiv 0 (\textit{mod } 2) \\ \frac{1}{3}(2^n + 1) & \textit{if } n \equiv 1 (\textit{mod } 2) \end{cases}$

2.2 Slim tree

The n^{th} slim tree $ST(n)$, $n \geq 2$ is recursively defined as follows and is 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.

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 $V=V_1\cup V_2\cup\{u\}$, $E=E_1\cup E_2\cup\{(u,u_1),(u,u_2),(r_1,l_2)\}$, $l=l_1$, $r=r_2$. See Figure 3(a).

Remark 2.7 *The Slim tree* $ST(n)$, $n\geq 3$, *contains* 2^{n-2} *number of vertex disjoint copies of* K_3, *where* K_3 *is the complete graph on three vertices.*

Remark 2.8 *Let* G *be the complete graph* K_3 *on three vertices. Then* $c(G)=1$.

Lemma 2.9 *Let* G *be a Slim tree* $ST(n)$, $n\geq 1$. *Then* $\nabla(G)\geq 2^{n-2}$

Proof. Every subgraph K_3 in $ST(n)$ is incident at a vertex common to all cycles in $ST(n)$ of length at least 4. Hence the maximum number of vertex disjoint cycles in G is the total number of K_3's in $ST(n)$. Therefore $c(G)\geq 2^{n-2}$. By Lemma 2.1, $\nabla(G)\geq 2^{n-2}$.

2.2.1 Algorithm Decycling Number of Slim tree $ST(n)$

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

Algorithm: Choose all the vertices in level $n-1$.

Output: $\nabla(ST(n))=2^{n-2}$.

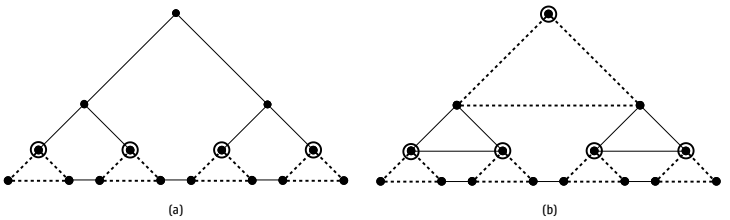

Figure 3: (a) darken cycles and circled vertices constitute cycle packing and minimum decycling set of *ST*(4)
(b) darken cycles and circled vertices constitute cycle packing and minimum decycling set of *XT*(4)

Proof of Correctness:

Let C be any cycle in $ST(n)$, $n\geq 2$. Then C contains at least one vertex in the $(n-1)^{th}$ level of $ST(n)$. Let S be the set of all vertices in level $n-1$. Removal of vertices in S ensures that it destroys all the cycles in $ST(n)$.

Theorem 2.10 *Let* G *be a Slim tree* $ST(n)$, $n\geq 1$. *Then* $\nabla(G)=2^{n-2}$

2.3 Christmas tree

The Christmas tree $CT(n)$ is composed of an nth slim tree $ST^l(n)=(V_1,E_1,u_1,l_1,r_1)$ and an $(n+1)th$ slim tree $ST^r(n)=(V_2,E_2,u_2,l_2,r_2)$. The node set of $CT(n)$ is $V_1\cup V_2$ and the edge set of $CT(n)$ is $E=E_1\cup E_2\cup\{(u_1,u_2);(l_1,r_2),(l_2,r_1)\}$.

Corollary 2.11 *Let G be a Christmas tree $CT(n)$, $n\geq 1$. Then $\nabla(G)=3.2^{n-2}$*

Proof. Christmas tree $CT(n)$ contains vertex disjoint copies of slim tree $ST(n)$ and $ST(n+1)$. Therefore result follows from Theorem 2.10.

2.4 X-tree

An X-tree $XT(n)$ is obtained from complete binary tree on $2^{n+1}-1$ vertices and adding paths P_i of length 2^i-1, through all the vertices at level i, for left to right, $1\leq i\leq n$. See Figure 3(b).

Remark 2.12 *The X-tree $XT(n)$, $n\geq 3$ contains 2^{n-2} number of vertex disjoint copies of K_3 induced by vertices in pairs of levels i and $i-1$, $i=n,n-3,\ldots$ 2 or 1 as the case may be.*

Lemma 2.13 *Let G be a X-tree $XT(n)$, $n\geq 1$. Then* $\nabla(G)\geq\begin{cases}\frac{1}{3}(2^{n+1}-2) & if\ n\equiv 0(mod\ 2)\\ \frac{1}{3}(2^{n+1}-1) & if\ n\equiv 1(mod\ 2)\end{cases}$.

Proof. We prove the result by induction on n.

Case (i): $n\equiv 0(mod\ 2)$

There are three cycles in $XT(2)$ and two of them share a common vertex. Hence $c(G)=2$ proving the result when $n=2$. Assume the result to be true for $n=2k$, $k\geq 1$. Consider $XT(2k+2)$. By Remark 2.11, there are 2^{2k+1} vertex disjoint copies of K_3 in $XT(2k+2)$ whose deletion disconnects the graph into $XT(2k)$ and a path on 2^{n-1} vertices.

Therefore by Remark 2.7, $c(XT(2k+2))\geq c(XT(2k))+2^{2k+1}=\frac{1}{3}(2^{2k+1}-2)+2^{2k+1}=\frac{1}{3}(2^{2k+3}-2)$.

Case (ii): $n\equiv 1(mod\ 2)$

When $n=1$, the result is trivial. Assume the result to be true for $n=2k-1$, $k\geq 1$.Consider $XT(2k+1)$. By Remark 2.11, there are 2^{2k} vertex disjoint copies of K_3 in $XT(2k+1)$ whose deletion disconnects the graph into $XT(2k-1)$ and a path on 2^{n-1} vertices. Therefore by Remark 2.7, $c(XT(2k+1))\geq c(XT(2k-1))+2^{2k}=\frac{1}{3}(2^{2k}-1)+2^{2k}=\frac{1}{3}(2^{2k+2}-1)$.

In both cases, the result follows from Lemma 2.1.

2.4.1 Algorithm Decycling Number of X-tree $XT(n)$

Input: $XT(n)$, $n \geq 3$

Algorithm:

(i) For n even, choose all the vertices in level $n-1, n-3, \ldots, 4, 2$.

(ii) For n odd, choose all the vertices in level $n-1, n-3, \ldots, 3, 1$.

Output: $\nabla(XT(n)) = \begin{cases} \frac{1}{3}(2^{n+1}-2), n \text{ even} \\ \frac{1}{3}(2^{n+1}-1), n \text{ odd} \end{cases}$

Proof of Correctness:

Let C be any cycle in $XT(n)$, $n \geq 2$. Then C contains atleast one vertex in level i of $XT(n)$, where $i \in \{n-1, n-3, \ldots, 2\}$ for n even and $i \in \{n-1, n-3, \ldots, 1\}$ for n odd. Let S be the set of all vertices in levels $n-1, n-3, \ldots, 4, 2$, when n is even and S be the set of all vertices in levels $n-1, n-3, \ldots, 3, 1$, when n odd. Removal of vertices in S ensures that it destroys all the cycle in $XT(n)$.

Theorem 2.14 *Let G be a X-tree $XT(n)$, $n \geq 1$. Then* $\nabla(G) = \begin{cases} \frac{1}{3}(2^{n+1}-2) & \text{if } n \equiv 0 (mod\ 2) \\ \frac{1}{3}(2^{n+1}-1) & \text{if } n \equiv 1 (mod\ 2) \end{cases}$.

2.5 1-rooted sibiling tree

1-rooted sibiling tree ST_n^1 is obtained from the 1-rooted complete binary tree T_n^1 by adding edges (sibiling edges) between left and right children of the same parent node. The k-rooted sibiling tree ST_n^k is obtained by k vertex disjoint 1-rooted sibiling tree ST_n^1 on 2^n vertices with roots say $r_1, r_2, \ldots, r_k$ and adding edges (r_i, r_{i+1}), $1 \leq k \leq (k-1)$. The diameter of ST_n^k is $2n+k-1$. See Figure 4.

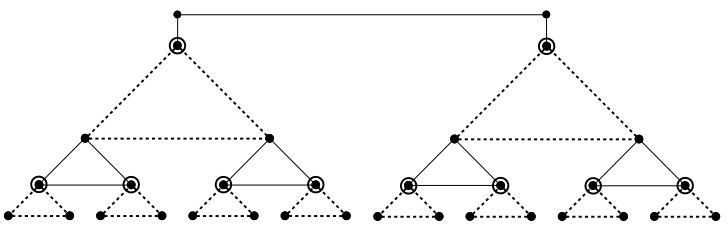

Figure 4: darken cycles and circled vertices constitute cycle packing and minimum decycling set of ST_4^2.

Proceeding along the same lines as for X-Tree, we have the following results.

Theorem 2.15 *Let G be a k-rooted sibiling tree ST_n^k, $n \geq 1$.* $\nabla(G) = \begin{cases} \frac{k}{3}(2^{n+1}-2) & \text{if } n \equiv 0 (mod\ 2) \\ \frac{k}{3}(2^{n+1}-1) & \text{if } n \equiv 1 (mod\ 2) \end{cases}$

2.6 l-siblings tree

The ST_n^k be a rooted sibling tree, $n \geq 1$. A graph which is obtained from two copies of rooted sibling tree ST_n^k, say ST_1^k, ST_2^k by joining each vertex in the last level (i.e., $(n-1)^{th}$ level) of ST_1^k with the corresponding vertex of ST_2^k by an edge is called the l-sibling tree and its denoted by l-ST_n^k. See Figure 5.

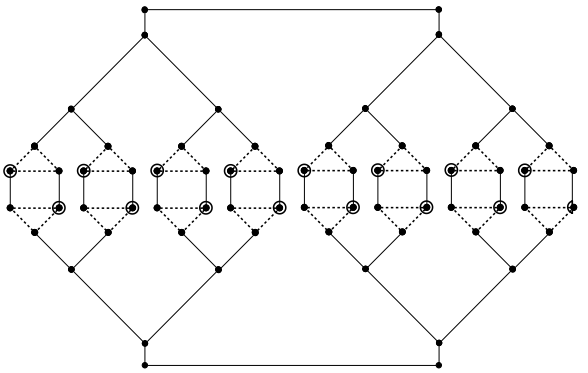

Figure 5: darken cycles and circled vertices constitute cycle packing and minimum decycling set of 2-ST_4^2.

Theorem 2.16 *Let* G *be a* l*-sibiling tree* l-ST_n^k, $n \geq 1$. *Then* $\nabla(G) = k.2^{n-2}$.

2.7 l-complete binary tree

Let T_n be a complete binary tree, $n \geq 1$. A graph which is obtained from two copies of complete binary tree T_n, say T_1, T_2 by joining each vertex in the last level (i.e., $(n-1)^{th}$ level) of T_1 and the corresponding vertex of T_2 is called the l-complete binary tree and its denoted by l-T_n. See Figure 2(b).

Remark 2.17 *Number of vertices in* l-T_n *is 3.* $2^{n-1}-2$, $n \geq 1$.

Proof. Since the graph l-T_n is obtained from two copies of T_n, the total number of vertices in l-T_n is equal to $2.2^{n+1}-2-2^n$=3. $2^{n-1}-2$.

Remark 2.18 *Number of edges in* $l-T_n$ *is* $2^{n+1}-4$, $n \geq 1$.

Theorem 2.19 *Let* G *be a* l*-complete binary tree* l-T_n, $n \geq 1$. *Then* $\nabla(G) = 2^{n-1}$.

2.8 k -dimensional generalized binary fat tree

The k -dimensional generalized binary fat tree $BFT(k)$ has $n = 2^k(k+1)$ nodes arranged in $k+1$ levels of 2^k nodes each. Each node has a distinct label $\langle z, j \rangle$, where j is the level of the node $(0 \le j \le k)$ and $z = (a_k \ldots a_j \ldots a_2 a_1)$ is a k bit binary number. Two nodes $\langle z, j \rangle$ and $\langle z', j' \rangle$ are adjacent if $j' = j+1$ and either w and w' are identical or $z' = (a_k \ldots \overline{a_j} \ldots a_2 a_1)$. The edges in the network are undirected. The nodes on level 0 are called the input nodes or just inputs of the network, and the nodes on level k are called the output nodes or just outputs. See Figure 6(a).

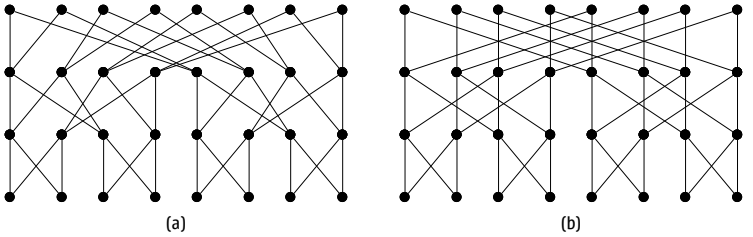

Figure 6: (a) 3-dimensional Generalized Binary Fat tree BFT(3); (b) 3-dimensional Butterfly Network BF(3)

Definition 2.20 *The k -dimensional butterfly $BF(k)$ has $n = 2^k(k+1)$ nodes arranged in $k+1$ levels of 2^k nodes each. Each node has a distinct label $\langle w, i \rangle$ where i is the level of the node $(0 \le i \le m)$ and w is a k -bit binary number that denotes the column of the node. All nodes of the form $\langle w, i \rangle$, $(0 \le i \le k)$, are said to belong to column w. Similarly, the i^{th} level L_i consists of all of the nodes $\langle w, i \rangle$, where w ranges over all k -bit binary numbers. Two nodes $\langle w, i \rangle$ and $\langle w', i' \rangle$ are linked by an edge if $i' = i+1$ and either w and w' are identical or w and w' differ only in the bit in position i'. (The bit positions are numbered 1 through k, the most significant bit being numbered 1). The edges in the network are undirected. The nodes on level 0 are called the input nodes or just inputs of the network, and the nodes on level k are called the output nodes or just outputs.* See Figure 6(b).

Rajasingh et al.[17] have proved that Figure 6(a) and Figure 6(b) are isomorphic.

Lemma 2.21 *Let G be an odd dimensional butterfly network $BF(n)$, n odd. Then $\nabla(G) \ge (n+1).2^{n-1}$, $n \ge 3$.*

Proof. The maximum number of vertex disjoint cycles between any pair of consecutive levels is 2^{n-1}. Since there are $n+1$ levels, there are maximum $(n+1)/2$ vertex disjoint levels. Hence $c(G) \ge (n+1).2^{n-2}$. By Lemma 2.1, $\nabla(G) \ge (n+1).2^{n-1}$.

2.8.1 Algorithm Decycling Number of Odd dimensional Butterfly network

Input: Butterfly network $BF(n)$, n odd and $n \geq 3$

Algorithm:

There are 2^k vertices in level i , i odd and $i < k$, split the vertices at level i into 2^{k-i-1} sets say S_i^j , $1 \leq j \leq 2^{k-i-1}$ consisting of 2^{i+1} from left to right. From each set, select the first 2^{i-1} vertices and last 2^{i-1} vertices in S_i and the singleton set, omitting 2^i vertices in the middle. See figure 8(b).

Output: $\nabla(BF(n)) = (n+1).2^{n-1}$

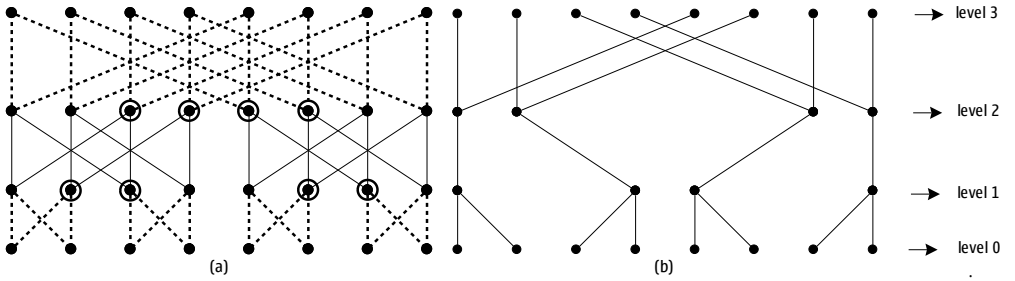

Figure 7: (a) darken cycles and circled vertices constitute cycle packing and minimum decycling set of *BF*(3)
(b) removed vertices constitute minimum decycling set of *BF*(3).

Proof of Correctness:

Since there are $(n+1).2^{n-2}$ vertex disjoint cycles in $BF(n)$. We need at least $(n+1).2^{n-2}$ many vertices to be removed, inorder to make the graph acyclic. Removal of these vertices ensures that it destroys all the cycle and induces a forest.

Theorem 2.22 *Let* G *be an odd dimensional butterfly network* $BF(n)$, n *odd. Then* $\nabla(G) = (n+1).2^{n-1}$, $n \geq 3$.

3 Conclusion

In this paper we have proved that $c(G) = \nabla(G)$ for hypertree, Christmas tree, slim tree, X -tree, k -rooted siblings tree, l -siblings tree, l -complete binary tree and k -dimensional generalized binary fat tree. The problem of decycling number of cube connected cycles, benes are under investigation.

4 References

[1] R. M. Karp, Reducibility among Combinatorial Problems, *Complexity of Computer Computations*, The IBM Research Symposia Series (1972), 85-103.

[2] J. Kleinberg, A. Kumar, Wavelength conversion in optical networks, *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, (1999), 566-575.

[3] D. S. Johnson, Approximation algorithms for combinatorial problems, *Journal of Computer and System*

Sciences, **9** (1974), 256-278.

[4] C. Wang, E. L. Lloyd, M. L. Soffa, Feedback vertex sets and cyclically reducible graphs, *Journal of the ACM*, **32** (1985), 296-313.

[5] A. Silberschatz, P.B. Galvin, G. Gagne, Operating systems concepts, *6th edition Wiley Publishers*, New york **2003**.

[6] D. Peleg, Size bounds for dynamic monopolies, *Discrete Applied Mathematics*, **86** (1998), 263-273.

[7] D. Peleg, Local majority voting, small coalitions and controlling monopolies cn graphs: a review, *Theoretical Computer Science*, **282** (2002), 231-257.

[8] R. Bar-Yehuda, D. Geiger, J. Naor, R. M. Roth, Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and Bayesian inference, *SIAM Journal on Computing*, **27** (1998), 942-959.

[9] P. Festa, P. M. Pardalos, M. G. C. Resende, Feedback set problems, *Du D-Z, Pardalos PM (edition) Handbook of combinatorial optimization*, supplement A. Kluwer Academic, Dordrecht **2000**, 209-259.

[10] H. L. Bodlaender, On disjoint cycles, *International Journal of Foundations of Computer Science*, **5** (1994), 59-68.

[11] A. Caprara, A. Panconesi, R.Rizzi, Packing cycles in undirected graphs, *Journal of Algorithms*, **48** (2003), 239-256.

[12] P. Biro, D. F. Manlove, R.Rizzi, Maximum weight cycle packing in Directed graphs, with Application to Kidney exchange programs, *Discrete Mathematics, Algorithms and Applications*, **1** (2009), 499.

[13] H. Chang, H. L. Fu, M. Y. Lien, The decycling number of outerplanar graphs, *Journal of Combinatorial Optimization*, **25** (2013), 536-542.

[14] C. H. Tsai, Linear array and ring embeddings in conditional faulty hypercubes, *Theoretical Computer Science*, **314** (2004), 431-443.

[15] J. R. Goodman and C. H. Sequin, A multiprocessor interconnection topology, *IEEE Transactions on Computers*, **30** (1981), 923-933.

[16] R. S. Rajan, J. Gopal, I. Rajasingh, T. M. Rajalaxmi and N. Parthiban, Combinatorial Properties of Root-fault Hypertrees, *Proceedia Computer Science*. (Communicated).

[17] I. Rajasingh, B. Rajan, R. S. Rajan, P. Manuel, Embedding in Fat Trees, *Journal of Combinatorial Mathematics and Combinatorial Computation*, **79** (2011), 139-146.