Computer Science
Technical Report

Colorado State University

A New Family of Cayley Graph Interconnection Networks of Constant Degree Four

Premkumar Vadapalli* **and Pradip K Srimani**
Department of Computer Science
Colorado State University
Ft. Collins, CO 80523

Technical Report CS-95-107

Computer Science Department
Colorado State University
Fort Collins, CO 80523-1873

Phone: (970) 491-5792 Fax: (970) 491-2466
WWW: http://www.cs.colostate.edu

*Premkumar Vadapalli is presently with Tartan Laboratories, Pittsburgh, Pennsylvania

A New Family of Cayley Graph Interconnection Networks of Constant Degree Four

Premkumar Vadapalli*and Pradip K Srimani
Department of Computer Science
Colorado State University
Ft. Collins, CO 80523

Abstract

We propose a new family of interconnection networks that are Cayley graphs with constant node degree 4. These graphs are regular, have logarithmic diameter and are maximally fault tolerant. We investigate different algebraic properties of these networks (including fault tolerance) and propose optimal routing algorithms. As far as we know, this is the first family of Cayley graphs of constant degree 4.

1 Introduction

Design of interconnection networks is an important integral part of any parallel processing or distributed system. Performance of the distributed system is significantly determined by the choice of the network topology. A very efficient interconnection network is the well-known binary n-cubes or hypercubes; they have been used to design various commercial multiprocessor machines and they have been extensively studied. For the past several years, there has been a spurt of research on a class of graphs called Cayley graphs that are very suitable for designing interconnection networks. Cayley graphs are based on permutation groups and include a large number of families of graphs, like star graphs [AK89, AK87, QMA91], hypercubes [BA84], pancake graphs [AK89, QAM93] and others [AT91, Sch91, DT92]. These graphs are symmetric (edges are bidirectional), regular, and seem to share many of the desirable properties like low diameter, low degree, high fault tolerance etc. with the well-known hypercubes (which are also Cayley graphs). An excellent survey of these Cayley graphs (along with extensive bibliography) can be found in [LJD93]. All Cayley graphs are regular, i.e., each node has the same degree, but for Cayley graphs studied to date, the degree of the nodes increases with the size of the graph (the number of nodes) either logarithmically or sublogarithmically. From a VLSI design point of view, these interconnection topologies are not so suitable for area efficient layout; we need constant degree networks (where the degree of a node does not change with the size of the network). There are also important applications for which the computing nodes in the interconnection network can have only a fixed number of I/O ports [SP89, CAB93]. There are graphs in the literature that have constant node degrees, like the Cube-Connected Cycles [PV81] where the degree of any node is 3 irrespective of the size of the graph. These cube-connected cycle graphs can be viewed as Cayley graphs [CCSW85]. There also exist graph topologies in the literature that have *almost* constant node degrees like the De

*Premkumar Vadapalli is presently with Tartan Laboratories, Pittsburgh, Pennsylvania

Bruijn graphs [PR82] or the Moebius graphs [LS82]. Constant degree network graphs are of considerable practical importance since De Bruijn graphs are being used for designing a 8096 node multiprocessor at JPL for the Galileo project [Pra91]. But, neither De Bruijn graphs nor Moebius graphs are regular; none of them can be viewed as Cayley graphs. Also, these graphs have a low vertex connectivity of only 2 (i.e., fault tolerance is minimal in the sense that the graphs cannot tolerate more than one faulty node) although most of the nodes in those graphs have degree larger than 2 (most of the nodes in De Bruijn graphs have degree 4 and most of the nodes in Moebius graphs have degree 3).

Our purpose in the present paper is to propose a new family of degree 4 Cayley graphs. The proposed family of graphs is regular of degree 4 irrespective of the size of the graph, has a logarithmic diameter and has a vertex connectivity of 4, i.e., the graphs are maximally fault tolerant. Compared with cube-connected cycle graphs, the proposed graph has a higher vertex connectivity (hence higher fault tolerance) and it accomodates a larger number of nodes than cube-connected cycle graph for the same diameter. The proposed family of graphs is also interesting from a graph theoretic point of view and as an attractive alternative to De Bruijn graphs for VLSI implementation in terms of regularity and greater fault tolerance without additional cost. The rest of the paper is organized as follows. Section 2 introduces the new topology and section 3 deals with two routing schemes and establishes the diameter of the graphs. Section 4 investigates different algebraic properties of the graph, including the existence of Hamiltonian circuits, and section 5 is devoted to the fault tolerance properties of the graph. Section 6 concludes the paper.

2 Degree Four Cayley Graphs

Degree Four Cayley Graph G_n is defined as a graph on $n \times 2^n$ vertices for any integer n, $n \geq 3$; each vertex is represented by a circular permutation of n symbols in lexicographic order where each symbol may be present in either uncomplemented or complemented form. Let t_k, $1 \leq k \leq n$ denote the k-th symbol in the set of n symbols (we use English alphabets as symbols; thus for $n = 4$, $t_1 = a$, $t_2 = b$, $t_3 = c$ and $t_4 = d$). We use t_k^* to denote either t_k or $\bar{t}_k$. Thus, for n distinct symbols, there are exactly n different cyclic permutation of the symbols in lexicographic order and since each symbol can be present in either complemented or uncomplemented form, the vertex set of G_n (i.e. the underlying group Γ) has a cardinality of $n.2^n$ (for example, for $n = 3$, the number of vertices in G_3 is 24; abc, cab, $\bar{c}ab$ are valid nodes while acb or bac are not). Let I denote the *identity permutation* $t_1 t_2 \cdots t_n$. Since each node is some cyclic permutation of the n symbols in lexicographic order, then if $a_1 a_2 \cdots a_n$ denotes the label of an arbitrary node and $a_1 = t_k^*$ for some integer k, then for all i, $2 \leq i \leq n$, we have $a_i = t^*_{(k+i) \bmod n+1}$. The edges of G_n are defined by the following four generators in the graph:

$$
\begin{aligned}
g(a_1 a_2 \cdots a_n) &= a_2 a_3 \cdots a_n a_1 \\
f(a_1 a_2 \cdots a_n) &= a_2 a_3 \cdots a_n \bar{a}_1 \\
g^{-1}(a_1 a_2 \cdots a_n) &= a_n a_1 \cdots a_{n-1} \\
f^{-1}(a_1 a_2 \cdots a_n) &= \bar{a}_n a_1 \cdots a_{n-1}
\end{aligned}
$$

Example: For $n = 3$, the vertex set of G_3 (the underlying group Γ) is given by $\{abc,\ \bar{a}bc,\ a\bar{b}c,\ ab\bar{c},\ cab,\ \bar{c}ab,\ c\bar{a}b,\ ca\bar{b},\ bca,\ \bar{b}ca,\ b\bar{c}a,\ bc\bar{a},\ \bar{a}\bar{b}c,\ a\bar{b}\bar{c},\ \bar{a}b\bar{c},\ \bar{c}\bar{a}b,\ c\bar{a}\bar{b},\ \bar{c}a\bar{b},\ \bar{b}\bar{c}a,\ b\bar{c}\bar{a},\ \bar{b}c\bar{a},\ \bar{a}\bar{b}\bar{c},\ \bar{b}\bar{c}\bar{a},\ \bar{c}\bar{a}\bar{b}\}$ and the generator set Ω is given by $\Omega = \{bc\bar{a},\ \bar{c}ab,\ cab,\ bca\}$. Note that since we are considering groups of permutation of distinct symbols which may be complemented or uncomplemented, the generators themselves are also permutations of complemented or uncomplemented symbols. Figure 1 shows the proposed degree four Cayley graph G_3 of dimension 3.

Remark 1

- *The set of four generators, $\Omega = \{f, g, f^{-1}, g^{-1}\}$ closed under inverse; in particular g is inverse of g^{-1} and f is inverse of f^{-1}; thus the edges in G_n are bidirectional.*
- *For an arbitrary n, $n > 2$, for any arbitrary node v of the graph G_n, $\delta(v) \neq v$ where $\delta \in \{g, f, g^{-1}, f^{-1}\}$.*

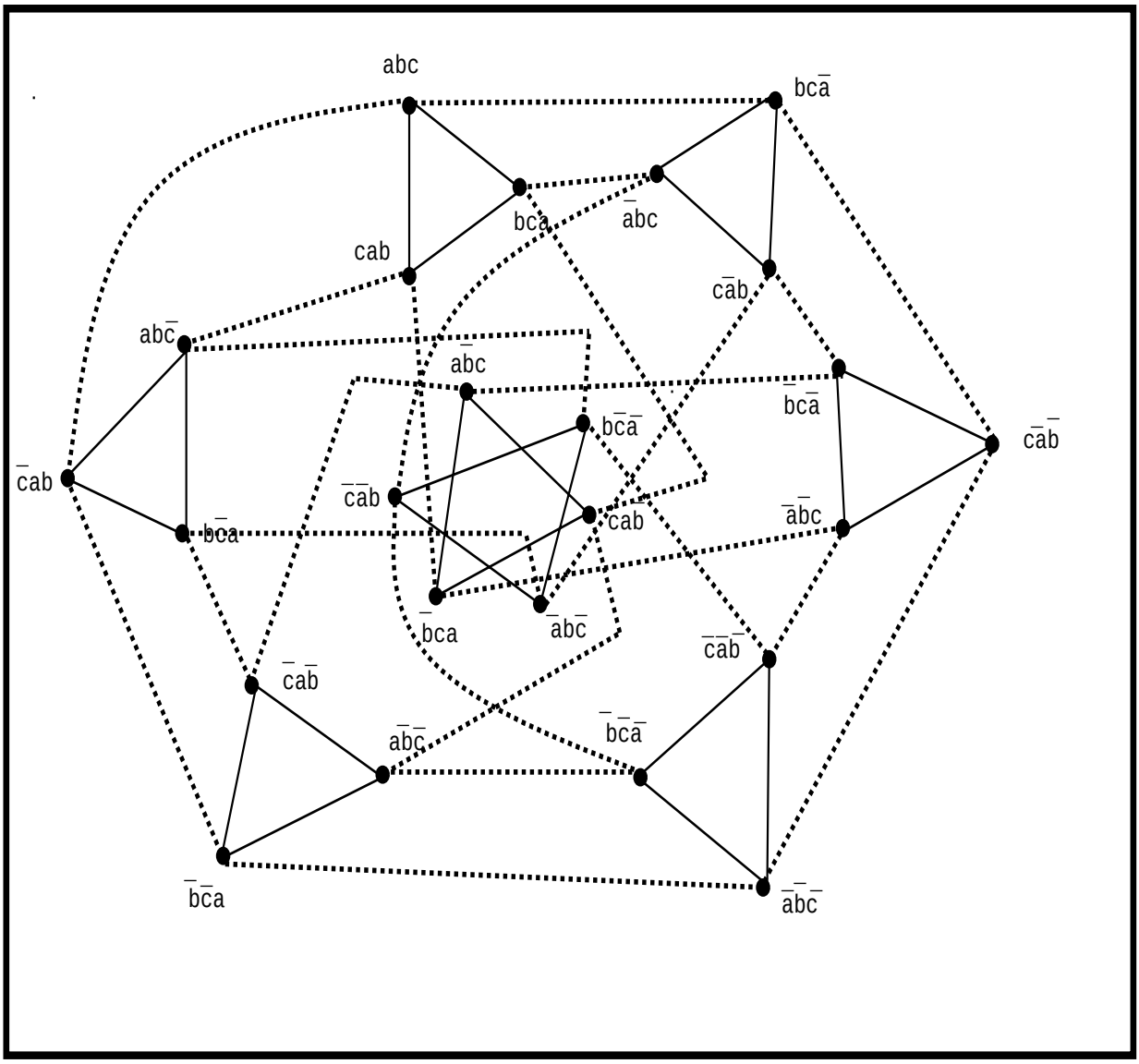

Figure 1: Example Graph for $n = 3$ (24 nodes)

Theorem 1 *For any n, $n \geq 3$, the graph G_n: (1) is a symmetric (undirected) regular graph of degree 4; (2) has $n.2^n$ vertices; and (3) has $n.2^{n+1}$ edges.*

Proof : (1) follows from the remarks 1 and 2 above. (2) follows from the fact that there are exactly n circular permutations of n symbols in lexicographic order (permutation with same cyclic ordering) and in any permutations each symbol can be either in complimented or uncomplemented form. (3) follows from (1) and (2). □

3 Routing Schemes

3.1 A Simple Routing Scheme

Since G_n is a Cayley graph, it is vertex symmetric [AK89], i.e., we can always view the distance between any two arbitrary nodes as the distance between the source node and the identity permutation by suitably renaming the symbols representing the permutations. For example, let $\bar{c}ab$ be the source node and $bc\bar{a}$ be the destination node. We can map the destination node to the identity node abc by renaming the symbols as $b \mapsto a$, $\bar{b} \mapsto \bar{a}$, $c \mapsto b$, $\bar{c} \mapsto \bar{b}$, $\bar{a} \mapsto c$ and $a \mapsto \bar{c}$. Under this mapping the source node becomes $\bar{b}\bar{c}a$. Then the paths between the original source and destination nodes become isomorphic to the paths between the node $\bar{b}\bar{c}a$ and the identity node abc in the renamed graph. Thus, in our subsequent discussion about a path from a source node to a destination node, the destination node is always assumed to be the identity node I without any loss of generality. The following algorithm S_R computes a path from an arbitrary source node $a_1a_2\cdots a_n$ in G_n to the identity node I.

Step 1: Compute k, $1 \leq k \leq n$, such that $a_k = t_n^*$.

Step 2: **If** $k > \lfloor \frac{n}{2} \rfloor$ **then** go along successive g^{-1} edges $(n-k)$ times
else go along successive g edges k times.

Step 3: **for** $i = 1$ **to** n **do**
go to the node $t_i^* t_{i+1}^* \cdots t_n^* t_1 t_2 \cdots t_i$ by either the g or the f edge;

Theorem 2 *The algorithm S_R correctly computes a path from an arbitrary node $a_1a_2\cdots a_n$ to the identity node I.*

Proof : The integer k always exists (Step 1) since the node $a_1a_2\cdots a_n$ is a cyclic permutation of the lexicographic ordering of n symbols (each symbol is in complemented or uncomplemented form). At the end of step 2, we reach a node $a_{k+1}a_{k+2}\cdots a_n a_1 \cdots a_k \equiv t_1^* t_2^* \cdots t_n^*$. At the beginning of step 3, we are at a node $t_1^* t_2^* \cdots t_n^*$ and we go to the node $t_2^* \cdots t_n^* t_1$ in the first iteration by taking either the g or the f edge (g edge if $t_1^* = t_1$ and the f edge if $t_1^* = \bar{t}_1$). Such a move obviously exists for each iteration and at the end of step 3 we have reached the node $t_1 \cdots t_n$, which is the destination. □

Theorem 3 *For an arbitrary node $a_1a_2\cdots a_n$ in G_n, the algorithm S_R generates a path of length $\leq \lfloor \frac{3n}{2} \rfloor$.*

Proof : If $k > \lfloor \frac{n}{2} \rfloor$ then $(n-k) \leq \lfloor \frac{n}{2} \rfloor$ and hence at the end of step 2 we reach the node $a_{k+1}a_{k+2}\cdots a_n a_1 \cdots a_k$ by a path of length $\leq \lfloor \frac{n}{2} \rfloor$. Step 3 generates a path of length exactly n from this intermediate node to the destination node. Hence the result. □

Corollary 1 *The upper bound on the diameter of G_n is given by $\mathcal{D}(G_n) \leq \lfloor \frac{3n}{2} \rfloor$.*

Remark 2 *Although the above algorithm does generate a path of length $\leq \lfloor \frac{3n}{2} \rfloor$ from an arbitrary node to the identity node, it does not generate the optimal (shortest) path in most cases.*

3.2 Diameter and Optimal Routing

Definition 1 *Consider an arbitrary node $s = a_1a_2\cdots a_n$ in G_n. There exists an unique integer k such that $a_k = t_1^*$. We define* **left distance** $D_L(s)$ *and* **right distance** $D_R(s)$ *of the node s (from the identity node) as follows:*

$$D_L(s) = 2(k - m_1 - 1) + (n - k + 1)$$

$$D_R(s) = 2(n - k - m_2) + (k - 1)$$

where

$$m_1 = \max_m\{\exists(i,j) \mid (1 \le j \le k) \wedge (1 \le i \le n - m + 1) \wedge a_ja_{j+1}\cdots a_{j+m-1} = t_it_{i+1}\cdots t_{i+m-1}\}$$

$$m_2 = \max_m\{\exists(i,j) \mid (k \le j \le n) \wedge (1 \le i \le n - m + 1) \wedge a_ja_{j+1}\cdots a_{j+m-1} = t_it_{i+1}\cdots t_{i+m-1}\}$$

Then, we define the distance of the node s (from the identity node) as

$$D(s) = \min\{D_L(s), D_R(s)\}$$

Note that the process of optimal routing from an arbitrary node to the identity node is equivalent to sorting a given permutation using the available operators (generators). The algorithm S_R does reach the identity node by simply constructing the identity node; it does not take into account if the source node is already partially sorted. That's why it is not optimal most of the time. Also, the generators f and f^{-1} provide complemenation of the symbols while the generators g and g^{-1} provide circular shifts without complementation. The parameters m_1 and m_2 measure the longest sequence of uncomplemented symbols on left and right of the special symbol t_1 (it is special because it occupies the first position in the destination identity node I). Also, the generators f and g provide left circular shifts while the generators f^{-1} and g^{-1} provide right circular shifts. The left (right) distance of a node attempts to compute the number of hops necessary to reach I by recognizing the *presortedness* on the left (right) of the special symbol t_1. The significance of these two distances will be clearer when we describe the optimal routing algorithm later.
Example 1: Consider the node $s = f\bar{g}hija\bar{b}cd\bar{e}$ in G_{10} (the identity node is $abcdefghij$). Here, $k = 6$, since $a_6 =$ "a" $= t_1$, $m_1(s) = 3$ (due to the substring "hij"), $m_2(s) = 2$ (due to the substring "cd"), $D_L(s) = 9$ and $D_R(s) = 11$. Hence $D(s) = 9$.

Theorem 4 *For an arbitrary node $s = a_1a_2\cdots a_n$ in G_n, $D(s') = D(s) \pm 1$, where $s' = \delta(s)$ and $\delta \in \{g, f, g^{-1}, f^{-1}\}$.*

Proof : We need to consider the case of each operator separately; assume $\delta(s) = s'$ with corresponding m_1', m_2' and k'.
Case 1: ($\delta = g$) We have $k' = k - 1$ and there can be four possible changes in m_1 and m_2 (note that either m_1 or m_2 or both can change at most by 1).

Subcase A: $m_1' = m_1 - 1$ & $m_2' = m_2 + 1$. We get $D_L(s') = 2(k' - m_1' - 1) + n - k' + 1 = D_L(s) + 1$ and $D_R(s') = 2(n - k' - m_2' + 1) + k' - 1 = D_R(s) - 1$; or $D(s') = D(s) \pm 1$.
Subcase B: $m_1' = m_1 - 1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s) + 1$ and $D_R(s') = D_R(s) + 1$; or $D(s') = D(s) + 1$.
Subcase C: $m_1' = m_1$ & $m_2' = m_2 + 1$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) - 1$; or $D(s') = D(s) - 1$.
Subcase D: $m_1' = m_1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) + 1$; or $D(s') = D(s) \pm 1$.

Case 2: ($\delta = f$) We have $k' = k - 1$ and there can be three possible changes in m_1 and m_2 (note that either m_1 or m_2 or both can change at most by 1).

Subcase A: $m_1' = m_1 - 1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s) + 1$ and $D_R(s') = D_R(s) + 1$; or $D(s') = D(s) + 1$.
Subcase B: $m_1' = m_1$ & $m_2' = m_2 + 1$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) - 1$; or $D(s') = D(s) - 1$.
Subcase C: $m_1' = m_1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) + 1$; or $D(s') = D(s) \pm 1$.

Case 3:($\delta = g^{-1}$) We have $k' = k + 1$ and there can be four possible changes in m_1 and m_2 (note that either m_1 or m_2 or both can change at most by 1).

Subcase A: $m_1' = m_1 + 1$ & $m_2' = m_2 - 1$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) + 1$; or $D(s') = D(s) \pm 1$.
Subcase B: $m_1' = m_1$ & $m_2' = m_2 - 1$. We get $D_L(s') = D_L(s) + 1$ and $D_R(s') = D_R(s) + 1$; or $D(s') = D(s) + 1$.
Subcase C: $m_1' = m_1 + 1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) - 1$; or $D(s') = D(s) - 1$.
Subcase D: $m_1' = m_1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s) + 1$ and $D_R(s') = D_R(s) - 1$; or $D(s') = D(s) \pm 1$.

Case 4:($\delta = f^{-1}$) We have $k' = k + 1$ and there can be three possible changes in m_1 and m_2 (note that either m_1 or m_2 or both can change at most by 1).

Subcase A: $m_1' = m_1$ & $m_2' = m_2 - 1$. We get $D_L(s') = D_L(s) + 1$ and $D_R(s') = D_R(s) + 1$; or $D(s') = D(s) + 1$.
Subcase B: $m_1' = m_1 + 1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) - 1$; or $D(s') = D(s) - 1$.
Subcase C: $m_1' = m_1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s) + 1$ and $D_R(s') = D_R(s) - 1$; or $D(s') = D(s) \pm 1$.

□

Corollary 2 *For the identity node $I = t_1 t_2 \cdots t_n$ in G_n, $D(I) = 0$ and $D(\delta(I)) = 1$ for any δ, $\delta \in \{g, f, g^{-1}, f^{-1}\}$.*

Given an arbitrary node $s = a_1 a_2 \cdots a_n$, the values of k, m_1 and m_2 can easily be computed in one linear scan of the node label; while doing so, the values of j (refer definition 1) corresponding to m_1 and m_2 are also stored in integer variables, say j_L and j_R respectively. In example 1 above, the values of j_L and j_R are 3 and 8 respectively. Once this is done, $D_L(s)$, $D_R(s)$ and $D(s)$ can be computed and the algorithm **Opt_Rout**, given in Figure 2 can be used to generate a path of length $D(s)$ from the node s to the destination node I, the identity permutation.

Theorem 5 *For an arbitrary node $s = a_1 a_2 \cdots a_n$ in G_n, the function $D(s)$, given by definition 1, correctly gives the distance of the node s from the identity node $I = t_1 t_2 \cdots t_n$.*

Procedure Opt_Rout($s, m_1, m_2, j_L, j_R, D_L(s), D_R(s)$**)**
if $D_L(s) < D_R(s)$ **then**
begin
for $i = 1$ **to** $(j_L - 1)$ **do**
Move to the g neighbor of the current node.
for $i = 1$ **to** $(j_L - 1) + (n - k + 1)$ **do**
if the last symbol of the current node is uncomplemented
then move to the g^{-1} neighbor
else move to the f^{-1} neighbor
for $i = 1$ **to** $k - j_L - m_1$ **do**
Move to the g^{-1} neighbor of the current node.
for $i = 1$ **to** $k - j_L - m_1$ **do**
if the first symbol of the current node is uncomplemented
then move to the g neighbor
else move to the f neighbor
end
else
begin
for $i = 1$ **to** $n - j_R - m_2 + 1$ **do**
Move to the g^{-1} neighbor of the current node.
for $i = 1$ **to** $(n - j_R - m_2 + 1) + (k - 1)$ **do**
if the last symbol of the current node is uncomplemented
then move to the g neighbor
else move to the f neighbor
for $i = 1$ **to** $j_R - k$ **do**
Move to the g neighbor of the current node.
for $i = 1$ **to** $j_R - k$ **do**
if the first symbol of the current node is uncomplemented
then move to the g^{-1} neighbor
else move to the f^{-1} neighbor
end

Figure 2: Algorithm Opt_Rout($s, m_1, m_2, j_L, j_R, D_L(s), D_R(s)$)

Proof : Theorem 4 states that for an arbitrary node s and its immediate neighbor s' ($s' = \delta(s)$, where $\delta \in \{g, f, g^{-1}, f^{-1}\}$), we have $D(s') \geq D(s) - 1$. Thus, to reach the identity node I from the node s in the graph G_n at least $D(s)$ hops are necessary. From the definition 1 and the construction of the algorithm **Opt_Rout**, it is evident that the algorithm constructs a path of length $D(s)$ from any given node s to the identity node I. Thus the proof follows. □

Corollary 3 *The algorithm* **Opt_Rout** *is an optimal (shortest) routing algorithm for nodes in the graph G_n.*

Corollary 4 *For an arbitrary node $s = a_1 a_2 \cdots a_n$ in G_n ($s \neq I$), there exists a δ, $\delta \in \{g, f, g^{-1}, f^{-1}\}$ such that $D(\delta(s)) = D(s) - 1$.*

Proof : Obvious from theorem 5. □

Theorem 6 *The diameter of the graph G_n is given by $\mathcal{D}(G_n) = \lfloor \frac{3n}{2} \rfloor$.*

Proof : Consider the node $x = \bar{t}_{\lfloor \frac{n}{2} \rfloor + 1} \cdots \bar{t}_n \bar{t}_1 \cdots \bar{t}_{\lfloor \frac{n}{2} \rfloor}$. We see that $D(x) = \lfloor \frac{3n}{2} \rfloor$. This, coupled with corollary 1 establishes the result. □

4 Topological Properties of the Graph G_n

In this section we investigate different interesting structural properties of the proposed degree four Cayley graphs.

Definition 2 *Any cycle in G_n consisting of only the f-edges (induced by the symmetric functions f or f^{-1}) is called an f-cycle. Similarly, any cycle in G_n consisting of only the g-edges (induced by the symmetric functions g or g^{-1}) is called an g-cycle.*

Example: In Figure 1, the cycle $\{\bar{b}\bar{c}a, \bar{c}ab, abc, bc\bar{a}, c\bar{a}\bar{b}, \bar{a}\bar{b}\bar{c}, \bar{b}\bar{c}a\}$ is an f-cycle while the cycles $\{\bar{b}\bar{c}a, \bar{c}a\bar{b}, a\bar{b}\bar{c}, \bar{b}\bar{c}a\}$ and $\{\bar{b}\bar{c}\bar{a}, \bar{c}\bar{a}\bar{b}, \bar{a}\bar{b}\bar{c}, \bar{b}\bar{c}\bar{a}\}$ are g-cycles.

Definition 3 *The complement of any vertex $u = a_1 a_2 \cdots a_n$ in G_n is the vertex $\bar{u}$ obtained by complementing the symbols in u, i.e., $\bar{u} = \bar{a}_1 \bar{a}_2 \cdots \bar{a}_n$. For example, complement of the vertex $u = c\bar{a}\bar{b}$ in G_3 is the vertex $\bar{u} = \bar{c}ab$.*

Lemma 1 *For an arbitrary pair of nodes u and v in G_n such that $g(u) = v$, the complement nodes satisfy the same relation, i.e., $g(\bar{u}) = \bar{v}$.*

Proof : Consider an arbitrary node $u = a_1a_2 \cdots a_n$; then v is given by $v = g(u) = a_2a_3 \cdots a_na_1$. So, $g(\bar{u}) = g(\bar{a}_1\bar{a}_2 \cdots \bar{a}_n) = \bar{a}_2\bar{a}_3 \cdots \bar{a}_n\bar{a}_1 = \bar{v}$. □

Theorem 7 *All of the $n.2^n$ nodes of G_n of dimension n are partitioned into vertex disjoint f-cycles of length $2n$; number of f-cycles in G_n is 2^{n-1}.*

Proof : Consider an arbitrary node $v = a_1a_2 \cdots a_n$ in G_n. For any i, $i \geq 1$, let $f^i(v) = f(f^{i-1}(v))$, where $f^1(v) = f(v)$. It is easy to observe that $f^n(v) = \bar{v} = \bar{a}_1\bar{a}_2 \cdots \bar{a}_n$ and $f^{2n}(v) = v$. Also, $f^i(v) \neq f^j(v)$ for $1 \leq i, j \leq 2n$. Thus, from an arbitrary vertex v if the f function is repeatedly applied, a cycle of length $2n$ is traced in the graph G_n. That these f-cycles are vertex disjoint follows from the fact that f^{-1} exists (also note that $f(v_1) = f(v_2)$, if and only if $v_1 = v_2$). □

Corollary 5 *For any vertex v in G_n, both v and $\bar{v}$ belong to the same f-cycle.*

Remark 3

- *Consider the symbol set $\{t_1, t_2, \cdots, t_n\}$ for G_n. For all k, $1 \leq k \leq n$, each f-cycle in G_n has a unique node starting with t_k and another unique node starting with $\bar{t}_k$.*
- *For each f-cycle in G_n, the unique node starting with t_1 is called the* leader *node. Since there are n symbols and the leader nodes start with t_1, there are 2^{n-1}* leader *nodes in G_n which is equal to the number of f-cycles in G_n. Each f-cycle is characterized by its* leader *node. For example, the leader node of the f-cycle cited in example 1 is abc.*
- *Consider an arbitrary leader node $t_1t_2^*t_3^* \cdots t_n^*$ (of some f-cycle); each leader node maps to a $(n-1)$ bit binary number by assigning 0 if $t_i^* = \bar{t}_i$ and 1 if $t_i^* = t_i$ for $2 \leq i \leq n$. This gives us a convenient way to number all the 2^{n-1} f-cycles in G_n from f_0 to $f_{2^{n-1}-1}$. Again, in example 1, the leader node abc maps to the 2-bit number $11 = 3$ and the hence f-cycle is numbered as f_3.*

Theorem 8 *All of the $n.2^n$ nodes of G_n of dimension n are partitioned into vertex disjoint g-cycles of length n; number of g-cycles in G_n is 2^n.*

Proof : Similar to the proof of theorem 7. □

Theorem 9 *For any arbitrary vertex v in G_n such that $f(v) = u$ and $g(v) = w$, there exists a vertex x such that $g(x) = u$ and $f(x) = w$; furthermore, the nodes v, u, w and x are all distinct.*

Proof : Consider an arbitrary vertex $v = a_1a_2\cdots a_n$. Then $u = f(v) = a_2\cdots a_n\bar{a}_1$ and $w = g(v) = a_2\cdots a_na_1$. Choose the node x as $x == g^{-1}(u) = \bar{a}_1a_2\cdots a_n$. Thus, $g(x) = u$ and $f(x) = a_2\cdots a_na_1 = w$. That these four nodes are distinct are also obvious from the fact that the different symbols in the nodes are distinct. □

Remark 4 *Theorem 9 provides a method to connect two different f-cycles via g-edges in G_n (note that for an arbitrary node v, the nodes v and u belong to the same f-cycle and the nodes x and w belong to a different f-cycle.*

Definition 4 *Two f-cycles, say f_i and f_j, are said to be adjacent if there exists a vertex $v \in f_i$ and a vertex $u \in f_j$ such that $v = g(u)$ or $u = g(v)$.*

Theorem 10 *If two f-cycles f_1 and f_2 are adjacent, then there are four g-edges connecting f_1 and f_2.*

Proof : Since f_1 and f_2 are adjacent, assume there exists a vertex $v \in f_1$ such that $g(v) = u$, where $u \in f_2$. Let $f(v) = w$. Then by theorem 9 $g^{-1}(w) = y = f^{-1}(u)$; thus y is a node in f_2. Also, by the corollary 5, $\bar{v}, \bar{w} \in f_1$ and $\bar{u}, \bar{y} \in f_2$ and $g(\bar{v}) = \bar{u}$ and $g^{-1}(\bar{w}) = \bar{y}$ (by lemma 1). Thus there are four g-edges between the two f-cycles f_1 and f_2. □

Theorem 11 *Each f-cycle in G_n is adjacent to n different f-cycles.*

Proof : Consider an arbitrary f-cycle with the leader $v = a_1a_2\cdots a_n$, where $a_1 = t_1$. Now, $g(v) = a_2a_3\cdots a_na_1 = y_0$ and the node y_0 belongs to the f-cycle with leader $a_1\bar{a}_2\bar{a}_3\cdots\bar{a}_n$. Then, consider the nodes y_i, $1 \leq i < n$, such that $y_i = g(v_i)$ where $v_i = f^i(v)$. We have $y_i = g(a_{i+1}a_{i+2}\cdots a_n\bar{a}_1\bar{a}_2\cdots\bar{a}_i) = a_{i+2}a_{i+3}\cdots a_n\bar{a}_1\bar{a}_2\cdots\bar{a}_ia_{i+1}$; this node y_i belongs to a f-cycle with the leader $a_1a_2\cdots a_i\bar{a}_{i+1}a_{i+2}\cdots a_n$. Obviously, the nodes y_i, $0 \leq i < n$, belong to different f-cycles (they have different leaders) and hence any f-cycle is adjacent to n different f-cycles in G_n. □

Corollary 6 *Consider an f-cycle f_i for a given i, $0 \leq i < 2^{n-1}$; i is a $(n-1)$ bit binary number, say $b_{n-1}b_{n-2}\cdots b_0$. Then the f-cycle f_i or $f_{b_{n-1}b_{n-2}\cdots b_0}$ is adjacent to the following n f-cycles: $f_{\bar{b}_{n-1}\bar{b}_{n-2}\cdots\bar{b}_0}$, $f_{\bar{b}_{n-1}b_{n-2}\cdots b_0}$, $f_{b_{n-1}\bar{b}_{n-2}b_{n-3}\cdots b_0}$, $\cdots$, $f_{b_{n-1}b_{n-2}\cdots\bar{b}_0}$.*

Theorem 12 *The graph G_n, $n \geq 3$, has a Hamiltonian cycle.*

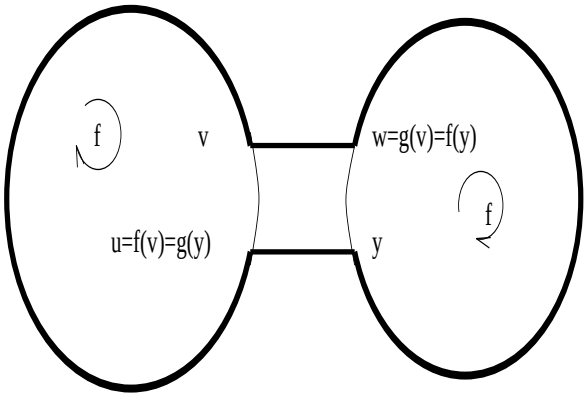

Figure 3: Combining two f-cycles to produce a larger cycle

Proof : Consider two arbitrary adjacent f-cycles, say f_1 and f_2. By theorem 9, there exist nodes $u, v \in f_1$ and nodes $w, y \in f_2$ such that $w = g(v) = f(y)$ and $u = f(v) = g(y)$. A larger cycle can be constructed involving all nodes of f_1 and f_2 by using these two g-edges as shown in Figure 3. This, coupled with the facts that f-cycles in G_n are vertex-disjoint and each f-cycle is adjacent to exactly n distinct other f-cycles, leads to the desired result. □

5 Fault Tolerance of G_n

The node fault tolerance of an undirected graph is measured by the vertex connectivity of the graph. A graph G is said to have a vertex connectivity ξ if the graph G remains connected when ar arbitrary set of less than ξ nodes are faulty. Obviously, the vertex connectivity of a graph G cannot exceed the minimum degree of a node in G; thus $\xi(G_n) \leq 4$ since G_n is a 4-regular graph for all values of n. A graph is called **maximally fault tolerant** if vertex connectivity of the graph equals the minimum degree of a node. Our purpose in this section is to show that the proposed graph G_n has a vertex connectivity of 4 and hence these graphs are maximally fault tolerant.

Definition 5 *For a given G_n compute the reduced graph RG_{n-1} in the following way: condense each f-cycle into a single node and label that node with the $(n-1)$ bit binary number corresponding to the f-cycle (see corollary 6); connect two arbitrary vertices by an undirected edge iff the corresponding f-cycles are adjacent in G_n.*

Remark 5

- *Figure 4 shows the reduced graph RG_2 and RG_3.*
- *Each vertex in RG_n has a binary label of length n and has a degree $(n+1)$ (each vertex in RG_n corresponds to a distinct f-cycle in G_{n+1}; each f-cycle in G_{n+1} has a n-bit binary label and is adjacent to $(n+1)$ distinct other f-cycles). Thus, RG_n is a $(n+1)$-regular undirected graph.*

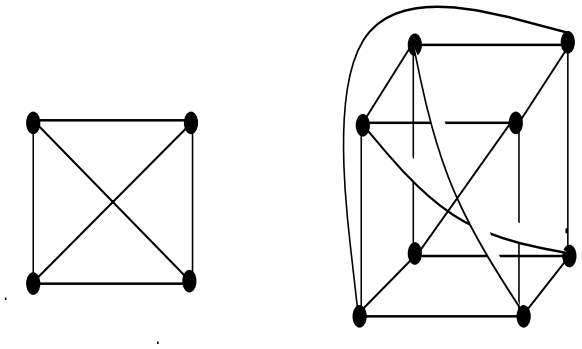

Figure 4: Reduced Graphs RG_2 and RG_3

- *Any two arbitrary vertices u and v in RG_n are connected by an edge iff either their Hamming distance is 1 or u is the complement of v by corollary 6.*
- *Consider a hypercube graph H_n of dimension n [SS88]; let $H_n = (V, E)$ where V is the set of 2^n vertices and E is the set of edges. Then H_n is always contained in the graph $RG_n = (V', E')$ such that $V' = V$ and $E' = E \cup \{(u, v) \mid (u, v \in V) \wedge (u = \bar{v})$.*

Theorem 13 *[SS88] A hypercube H_n has a vertex connectivity n, i.e. for any two given source and destination nodes, there are n vertex disjoint paths.*

Remark 6 *For any two given nodes u and v in H_n such that Hamming distance of u and v is k, there are k paths of length k which are obtained by keeping the similar bits of u and v the same throughout and changing the dissimilar bits selectively for different paths, and there $n - k$ paths of length $k + 2$ which are obtained by complementing one (different ones for different paths) similar bit, keeping it fixed and changing other bits and then complementing the similar bit again at the last step. See [SS88] for details.*

Theorem 14 *Vertex connectivity of RG_n is $(n + 1)$.*

Proof : Consider two arbitrary nodes $u = u_0u_1 \cdots u_{n-1}$ and $v = v_0v_1 \cdots v_{n-1}$ in RG_n. Since H_n is contained in RG_n, there are n vertex disjoint paths between u and v. We need to show the existence of the $(n+1)$-st vertex disjoint path between u and v. Let the Hamming distance of u and v be k. We need consider three cases:
Case 1: $k = n$. There n paths between u and v of length n (theorem 13). Now, in this case $u = \bar{v}$ and by definition there is a direct edge from u to v in RG_n.

Case 2: $k = n - 1$. Without loss of generality, let $u_0 = v_0$ and $u_i \neq v_i$ for $1 \leq i \leq n-1$. There are $n-1$ paths between u and v of length $n-1$ (theorem 13); all nodes involved has the same value u_0 in their 0-th bit positions. In RG_n, the other two vertex disjoint paths are computed as follows: (1) go from u to $u' = \bar{u}_0 u_1 \cdots u_{n-1}$ and from u' to v (there is an edge between u' and v since $u' = \bar{v}$); (2) go from v to $v' = \bar{v}_0 v_1 \cdots v_{n-1}$ and from v' to u (there is an edge between v' and u since $v' = \bar{u}$).

Case 3: $1 \leq k < n-1$, i.e., there are at least two bit positions where the source and the destination disagree. We already have n vertex disjoint paths: k of them are of length k (each node on all these paths have $n-k$ bits in the same state; there are k matching bits in the source and the destination node) and $n-k$ of them are of length $k+2$ (each node on all these paths has exactly one of the $n-k$ common bits inverted). To get the $(n+1)$st vertex disjoint path from u to v we do the following: go from u to $\bar{u}$, complement the k bits (the bit positions where u and v disagree) one at a time in any order to reach the node $\bar{v}$ and then go to v. Note that each node on this last path has $n-k$ (≥ 2) common bits inverted. □

Lemma 2 *Consider two arbitrary nodes u and v in G_n such that u and v belong to different f-cycles. Then there exist four vertex disjoint paths between u and v.*

Proof : Let $u \in f_i$ and $v \in f_j$ where $i \neq j$. Consider the following four f-cycles adjacent to f_i:

$$g(u) = u_1 \in f_{i1}, \quad g^{-1}(u) = u_2 \in f_{i2},$$

$$gf(u) = u_3 \in f_{i3}, \quad gf^{-1}(u) = u_4 \in f_{i4}$$

It is easy to see that for $n \geq 4$ these four f-cycles are distinct. Similarly, the node v can reach (by vertex disjoint paths) to four distinct f-cycles, say $f_{j1}, f_{j2}, f_{j3}, \; and \; f_{j4}$. Note that it is possible that $f_{i\ell} = f_{jk}$ for some ℓ and m. By Menger's theorem [Har72], given two sets of nodes V_1 and V_2 such that $|V_1| = |V_2| = n$ in a n-connected graph, there are n vertex disjoint paths connecting the nodes from V_1 to those in V_2. The reduced graph RG_{n-1} corresponding to G_n is n-connected by the previous theorem and hence for $n \geq 4$ there are 4 vertex disjoint paths connecting the two sets of 4 distinct f-cycles. Thus, there exist four vertex disjoint paths between u and v.

For the case $n = 3$, choose f_{i1}, f_{i2}, f_{i3} similarly but choose f_{i4} such that $f^{-1}(u) = u_4 \in f_{i4}$ (i.e., f_{i4} is the given f-cycle f_i. Choose $f_{j1}, f_{j2}, f_{j3}, \; and \; f_{j4}$ similarly. Note that in this case there are exactly 4 f-cycles in the entire graph G_3; thus, there is an one-to-one and onto mapping from one set of f-cycles to the other. Thus again, there exist four vertex disjoint paths between u and v. □

Lemma 3 *Consider two arbitrary nodes u and v in G_n such that u and v belong to the same f-cycle. Then there exist four vertex disjoint paths between u and v.*

Proof : Since u and v belong to the same f-cycle, we directly get two vertex disjoint paths between u and v along the given f-cycle. Consider the following four f-cycles:

$$g(u) = u_1 \in f_1, \quad g^{-1}(u) = u_2 \in f_2$$

$$g(v) = v_1 \in f_3, \quad g^{-1}(v) = v_2 \in f_4$$

Note that f_1 and f_2 are distinct and f_3 and f_4 are distinct; but one of f_1 and f_2 may be the same as one of f_3 and f_4 (when $u = f(v)\ or\ v = f(u)$). In any case, there are two vertex disjoint paths between the two sets of two f-cycles each by the n-connectivity of the reduced graph RG_{n-1} corresponding to G_n. Thus there exist four vertex disjoint paths between u and v. □

Theorem 15 *The graph G_n is 4-connected for any given n, $n \geq 3$.*

Proof : Obvious from the previous two theorems. □

Remark 7 *The graph G_n is maximally fault tolerant.*

Remark 8 *The proofs of the preceding two theorems outline a possible strategy to compute the vertex disjoint paths between two nodes in G_n, although that may not give four such shortest paths. Given the source node s and the destination node d, we can also run Dinic's algorithm for maximal flow assuming each edge has a unit capacity to compute the 4 node disjoint paths between s and d; see [ET75, Eve79] for details.*

6 Conclusion

In this paper we have proposed a new family of Cayley graphs with a constant degree of four. The graph has a diameter logarithmic in the number of nodes, are regular and dense, and is maximally fault tolerant. We have investigated different algebraic properties of the graph and proposed simple routing algorithms. The proposed graph seems to be an attractive alternative to the well known binary De Bruijn graphs for designing interconnection networks especially when the network is needed to be maximally fault tolerant. It is to be noted that while simple routing is very efficient for De Bruijn graphs [PR82], optimal routing in those graphs is complicated [GS86]; simple routing in the proposed graphs is very efficient and optimal routing, while more complicated than simple routing, is relatively simpler than that for De Bruijn graphs. When one compares the proposed graphs with the cube-connected cycles (only other Cayley graph of fixed degree), proposed graphs have two distinct advantages: vertex connectivity is one higher than that of cube-connected cycles (although at a higher cost in terms of edges) and they can accommodate much larger number of nodes for a given diammeter of the network. It is also to be

noted that although cube-connected cycles can be viewed as Cayley graphs [CCSW85] using complicated transformations, it is yet to be seen how Cayley graph theoretic tools can be utilized to establish topological properties of the cube-connected cycles. Further study of the proposed graph is needed to make it really useful in network design; most important of them are to to design fault tolerant routing algorithm for this graph as well as to study embedding of other graphs like hypercubes and meshes etc. Although one may easily get a fault tolerant routing algorithm in the line of the Remark 8 in a straightforward way, the resulting algorithm wouldn't be efficient; neither would it compute the shortest path in the remaining graph in presence of faults. Our investigations are underway and we intend to report further results soon.

Acknowledgement: The authors are grateful to the reviewers for many detailed comments that greatly improved the presentation.

References

[AK87] S. B. Akers and B. Krishnamurthy. The star graph: an attractive alternative to n-cube. In *Proceedings of International Conference on Parallel Processing (ICPP-87)*, pages 393–400, St. Charles, Illinois, August 1987.

[AK89] S. B. Akers and B. Krishnamurthy. A group-theoretic model for symmetric interconnection networks. *IEEE Transactions on Computers*, 38(4):555–566, April 1989.

[AT91] B. W. Arden and K. W. Tang. Representation and routing of Cayley graphs. *IEEE Transactions on Communications*, 39:1533–1537, December 1991.

[BA84] L. Bhuyan and D. P. Agrawal. Generalized hypercube and hyperbus structure for a computer netwrk. *IEEE Transactions on Computers*, 33(3):323–333, March 1984.

[CAB93] C. Chen, D. P. Agrawal, and J. R. Burke. dBCube: a new class of hierarchical multiprocessor interconnection networks with area efficient layout. *IEEE Transactions on Parallel and Distributed Systems*, 4(12):1332–1344, December 1993.

[CCSW85] G. E. Carlsson, J. E. Cruthirds, H. B. Sexton, and C. G. Wright. Interconnection networks based on a generalization of cube-connected cycles. *IEEE Transactions on Computers*, C-34(8):769–772, 1985.

[DT92] K. Day and A. Tripathi. Arrangement graphs: a class of generalized star graphs. *Information Processing Letters*, 42:235–241, July 1992.

[ET75] S. Even and R. E. Tarjan. Network flow and testing graph connectivity. *SIAM J. on Computing*, 4:507–518, 1975.

[Eve79] S. Even. *Graph Algorithms*. Pitman Publishing, 1979.

[GS86] S. Guha and A Sen. On fault tolerant distributor communication architecture. *IEEE Transactions on Computers*, C-35(3):281–283, March 1986.

[Har72] F. Harary. *Graph Theory*. Addison-Wesley, Reading, MA, 1972.

[LJD93] S. Lakshmivarahan, J. S. Jwo, and S. K. Dhall. Symmetry in interconnection networks based on Cayley graphs of permutation groups: a survey. *Parallel Computing*, 19:361–407, 1993.

[LS82] W. E. Leland and M. H. Solomon. Dense trivalent graphs for processor interconnection. *IEEE Transactions on Computers*, 31(3):219–222, March 1982.

[PR82] D. K. Pradhan and S. M. Reddy. A fault tolerant communication architecture for distributed systems. *IEEE Transactions on Computers*, C-31(9):863–870, September 1982.

[Pra91] D. K. Pradhan. Fault tolerant VLSI architectures based on de Bruijn graphs (Galileo in the mid nineties). *DIMACS Series in Discrete Mathematics*, 5, 1991.

[PV81] F. Preparata and J. Vuillemin. The cube-connected cycles: a versatile network for parallel computation. *Communications of ACM*, 24(5):30–39, May 1981.

[QAM93] K. Qiu, S. G. Akl, and H. Meijer. On some properties and algorithms for the star and pancake interconnection networks. *Journal of Parallel and Distributed Computing*, 1993.

[QMA91] K. Qiu, H. Meijer, and S. G. Akl. Decomposing a star graph into disjoint cycles. *Information Processing Letters*, 39(3):125–129, 1991.

[Sch91] I. D. Scherson. Orthogonal graphs for the construction of interconnection networks. *IEEE Transactions on Parallel and Distributed Systems*, 2(1):3–19, 1991.

[SP89] M. R. Samatham and D. K. Pradhan. The De Bruijn multiprocessor network: a versatile parallel processing and sorting network for VLSI. *IEEE Transactions on Computers*, 38(4):567–581, April 1989.

[SS88] Y. Saad and M. H. Shultz. Topological properties of hypercubes. *IEEE Transactions on Computers*, 37(7):867–872, July 1988.