Optimal Routing in Trivalent Cayley Graph Network[*]

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

Technical Report CS-96-114

[*]To appear in *Information Processing Letters*

Optimal Routing in Trivalent Cayley Graph Network[*]

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

Abstract

We develop an optimal routing algorithm as well as establish the actual value of the diameter of the newly proposed trivalent Cayley network graphs in [VS95]

1 Introduction

Fixed node degree networks constitute an important and essential component in designing any distributed system. Networks, in which degree of the nodes increases with the size of the network, are not suitable for applications involving large number of nodes [CAB93]. Fixed node-degree networks are also needed from VLSI implementation point of view [SP89]; there are applications where the computing nodes in the interconnection network can have only a fixed number of I/O ports [CAB93]. There are a few network graphs in the literature [PR82, LS82, PN93] where the node degree does not increase with network size; most popular among them are the De Bruijn graphs [PR82] or the Moebius graphs [LS82], but they are not regular and they are also not maximally fault tolerant. Very recently authors in [VS95] have proposed a new family of *regular* networks of degree 3; it has been shown that the network has a has a logarithmic diameter in the number of nodes and has a vertex connectivity of 3, i.e., the graphs are maximally fault tolerant (vertex connectivity cannot exceed the node degree). Although a simple routing algorithm was provided that gave an upper bound on the diameter of the network, optimal routing in the network was left as an open problem. Our purpose in the present paper is to develop an optimal routing algorithm as well as establish the exact value of the diameter of these trivalent Cayley network graphs.

2 Trivalent Cayley Graph G_n

In this section, we briefly recall the construction of the trivalent Cayley graphs G_n to facilitate our design of the optimal routing algorithm in the next section; details about the topological properties and fault tolerance of these graphs can be found in [VS95].

G_n is defined as a symmetric (undirected) graph on $N = n \times 2^n$ vertices for any integer $n, n \geq 2$; each vertex corresponds to a circular permutation of n symbols in lexicographic order where each symbol may be present in

[*]To appear in *Information Processing Letters*

either un complemented 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 (disregarding the complements) and since each symbol can be present in either complemented or un complemented form, the vertex set of G_n 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 $\bar{b}ac$ are not (since neither acb nor bac is a cyclic permutation of the three symbols in lexicographic order). 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 three generators in the graph:

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

Note that The three generators are closed under inverse; in particular g is its own inverse ($g = g^{-1}$) and f is inverse of f^{-1}; thus the edges in G_n are bidirectional. Also, For any n, $n \geq 2$, the graph G_n: (1) is a symmetric (undirected) regular graph of degree 3; (2) has $n \times 2^n$ vertices; and (3) has $3n \times 2^{n-1}$ edges. Figure 1 shows the trivalent Cayley graphG_3 of dimension 3.

3 Diameter and Optimal Routing

Since G_n is a Cayley graph [VS95], 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.

Definition 1 *Consider an arbitrary node $s = a_1 a_2 \cdots a_n$ 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:*

$$\begin{aligned} D_L(s) &= 2(k-1-m_1-1) + c_1 + 2(n-k+1) - c_2 && \textit{if } (j_L + m_1 < k) \\ &= 2(k-1-m_1) + c_1 + 2(n-k+1) - c_2 && \textit{otherwise} \\ D_R(s) &= 2(n-k+1-m_2-1) + c_2 + 2(k-1) - c_1 && \textit{if } (j_R + m_2 \leq n) \\ &= 2(n-k+1-m_2) + c_2 + 2(k-1) - c_1 && \textit{otherwise} \end{aligned}$$

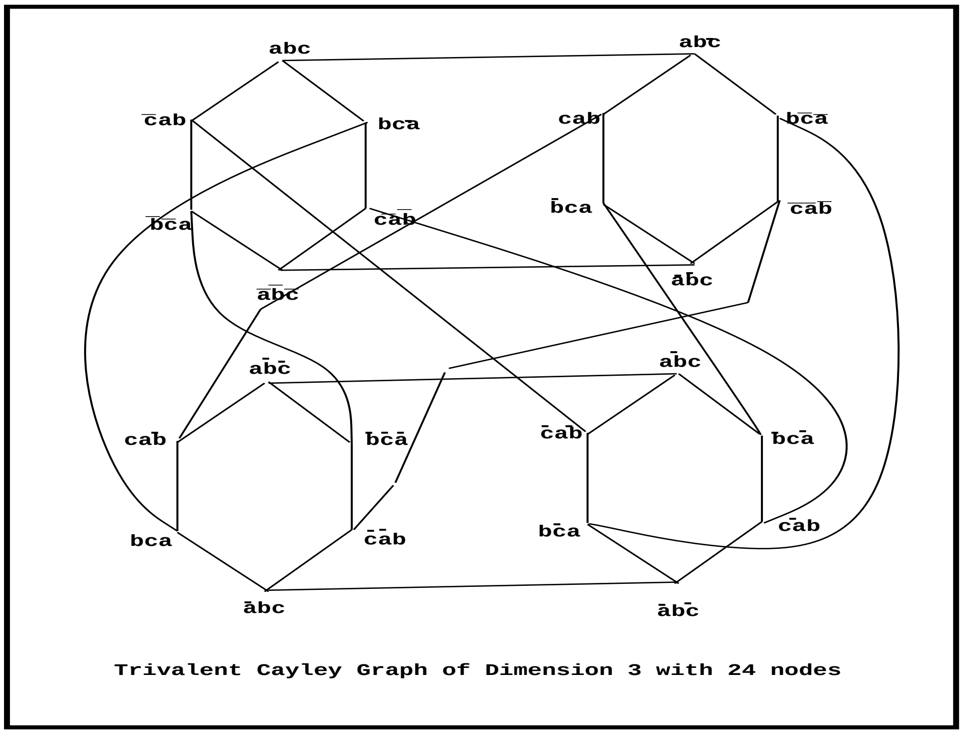

Figure 1: Trivalent Cayley Graph G_3

where

$$
\begin{aligned}
c_1 &= |\{i \mid 1 \le i < k \wedge a_i = \bar{t}_p, \text{for some p}\}| \\
c_2 &= |\{i \mid k \le i \le n \wedge a_i = \bar{t}_p, \text{for some p}\}| \\
m_1 &= \max_m\{\exists(i_L, j_L) \mid (1 \le j_L \le k) \wedge (1 \le i_L \le n-m+1) \wedge a_{j_L}a_{j_L+1}\cdots a_{j_L+m-1} = t_{i_L}t_{i_L+1}\cdots t_{i_L+m-1}\} \\
m_2 &= \max_m\{\exists(i_R, j_R) \mid (k \le j_R \le n) \wedge (1 \le i_R \le n-m+1) \wedge a_{j_R}a_{j_R+1}\cdots a_{j_R+m-1} = t_{i_R}t_{i_R+1}\cdots t_{i_R+m-1}\}
\end{aligned}
$$

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

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

Remarks:

- The symbol $a_k = t_1^*$ divides the node symbol sequence $a_1a_2\cdots a_n$ into two parts: $a_1a_2\cdots a_{k-1}$ and $a_ka_{k+1}\cdots a_n$. The parameter c_1 indicates the number of complemented symbols in the first part and the parameter c_2 indicates the number of complemented symbols in the second part.
- The parameter m_1 indicates the length of the longest substring(s) of un complemented symbols in the first part. We use j_L to indicate the left end of the leftmost of such substrings. If $m_1 = 0$, then we set $j_L = 1$.
- The parameter m_2 indicates the length of the longest substring(s) of un complemented symbols in the second part. We use j_R to indicate the left end of the leftmost of such substrings. If $m_2 = 0$, then we set $j_R = k$.

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"), $j_L = 3$, $m_2(s) = 2$ (due to the substring "cd"), $j_R = 8$, $c_1(s) = 1$(due to $\bar{g}$), $c_2(s) = 2$ (due to $\bar{b}$ and $\bar{e}$), $D_L(s) = 13$ (we use the second formula as we do not have any complemented symbol to the right of the string corresponding to m_1 before "a") and $D_R(s) = 15$ (we use the first formula as we have "$\bar{e}$" to the right of the string corresponding to m_2). Hence $D(s) = 13$.

Theorem 1 *For an arbitrary node* $s = a_1 a_2 \cdots a_n$ *in* G_n, $D(s') = D(s) \pm 1$, *where* $s' = \delta(s)$ *and* $\delta \in \{g, f, f^{-1}\}$.

Proof : We need to consider the case of each operator separately; assume $\delta(s) = s'$ with corresponding m'_1, m'_2,c'_1, c'_2 and k'.

Case 1: ($\delta = g$**)** We have $k' = k$, $m'_1 = m_1$, $c'_1 = c_1$ and there can be four possible changes in c_2 and m_2 (note that either c_2 or m_2 or both can change at most by 1).

> **Subcase A:** $c'_2 = c_2 - 1$ & $m'_2 = m_2 + 1$. We get $D_L(s') = D_L(s) + c_2 - c'_2 = D_L(s) + 1$ and $D_R(s') = 2(n - k' + 1 - m'_2) + c'_2 + 2(k' - 1) - c'_1$ = $D_R(s) - 1$; or $D(s') = D(s) \pm 1$.
> **Subcase B:** $c'_2 = c_2 - 1$ & $m'_2 = m_2$. We get $D_L(s') = D_L(s) + 1$ and $D_R(s') = D_R(s) - 1$; or $D(s') = D(s) \pm 1$.
> **Subcase C:** $c'_2 = c_2 + 1$ & $m'_2 = m_2 - 1$. 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 D:** $c'_2 = c_2 + 1$ & $m'_2 = m_2$. 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 four possible changes in c_1, c_2,m_1 and m_2

> **Subcase A:** $c'_1 = c_1 - 1$, $c'_2 = c_2$, $m'_1 = m_1$ & $m'_2 = m_2 + 1$. 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:** $c'_1 = c_1 - 1$, $c'_2 = c_2$, $m'_1 = m_1$ & $m'_2 = m_2$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) + 1$; or $D(s') = D(s) \pm 1$.
> **Subcase C:** $c'_1 = c_1$, $c'_2 = c_2 + 1$, $m'_1 = m_1$ & $m'_2 = m_2$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) \pm 1$($+$ if we have the same formula for $D_R(s)$ and $D_R(s')$;$-$ otherwise); or $D(s') = D(s) \pm 1$.
> **Subcase D:** $c'_1 = c_1$, $c'_2 = c_2 + 1$, $m'_1 = m_1 - 1$ & $m'_2 = m_2$. We get $D_L(s') = D_L(s) + 1$ and $D_R(s') = D_R(s) \pm 1$($+$ if we have the same formula for $D_R(s)$ and $D_R(s')$;$-$ otherwise); or $D(s') = D(s) \pm 1$.

Case 3:($\delta = f^{-1}$**)** We have $k' = k + 1$ and there can be four possible changes in c_1,c_2,m_1 and m_2

> **Subcase A:** $c'_1 = c_1 + 1$, $c'_2 = c_2$, $m'_1 = m_1$ & $m'_2 = m_2 - 1$. 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:** $c'_1 = c_1 + 1$, $c'_2 = c_2$, $m'_1 = m_1$ & $m'_2 = m_2$. We get $D_L(s') = D_L(s) + 1$ and

$D_R(s') = D_R(s) - 1$; or $D(s') = D(s) \pm 1$.

Subcase C: $c_1' = c_1, c_2' = c_2 - 1, m_1' = m_1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s)+1$ and $D_R(s') = D_R(s) \mp 1(-$ if we have the same formula for $D_R(s)$ and $D_R(s')$;+ otherwise); or $D(s') = D(s) \pm 1$.

Subcase D: $c_1' = c_1$, $c_2' = c_2 - 1$, $m_1' = m_1 + 1$ & $m_2' = m_2$. We get $D_L(s') = D_L(s) - 1$ and $D_R(s') = D_R(s) \mp 1(-$ if we have the same formula for $D_R(s)$ and $D_R(s')$;+ otherwise); or $D(s') = D(s) \pm 1$.

□

Corollary 1 *For the identity node $I = t_1t_2 \cdots t_n$ in G_n, $D(I) = 0$ and $D(\delta(I)) = 1$ for any δ, $\delta \in \{g, f, f^{-1}\}$.*

Remark: Given an arbitrary node $s = a_1a_2 \cdots a_n$, the values of k, c_1, c_2, m_1 and m_2 can be easily computed in linear time by scanning the node label once; while doing so, the values of j_L and j_R (refer definition 1 and the following remarks) corresponding to m_1 and m_2 respectively may also be stored as integer variables. 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 according to definition 1 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 2 *Given an arbitrary node s in G_n, the algorithm Opt_Rout correctly generates an optimal (shortest) path of length D(s) from the node s to the identity node I.*

Proof : From the definition 1 and the construction of the algorithm, it is evident that the algorithm constructs a path of length $D(s)$ for a given node s. From theorem 1, application of any one of the three operators $g, f,$ andf^{-1} can decrement the value of D of any node at best by one and since these are the only edges in the graph, the algorithm does generate the optimal path from the node s to the destination node I. □

Corollary 2 *For an arbitrary node $s = a_1a_2 \cdots a_n$ in G_n, the function $D(s)$, given by definition 1, correctly gives the distance of the node s from the identity node $I = t_1t_2 \cdots t_n$.*

Corollary 3 *For an arbitrary node $s = a_1a_2 \cdots a_n$ in G_n ($s \neq I$), there exists a δ, $\delta \in \{g, f, f^{-1}\}$ such that $D(\delta(s)) = D(s) - 1$.*

Proof : Obvious from theorem 2. □

Theorem 3 *The diameter of the graph G_n is given by*

$$\mathcal{D}(G_n) = \begin{cases} 2n-2 & \textit{if } n \textit{ is even} \\ 2n-3 & \textit{if } n \textit{ is odd} \end{cases}$$

```
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 $f$ 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 un complemented
                then move to the $g$ neighbor and then to the $f^{-1}$ neighbor
                else move to the $f^{-1}$ neighbor
        if $(k - j_L - m_1) > 0$ then
            begin
                for $i = 1$ to $k - j_L - m_1 - 1$ do
                    Move to the $f^{-1}$ neighbor of the current node.
                Move to the $g$ neighbor of the current node.
                for $i = 1$ to $k - j_L - m_1 - 1$ do
                    if the first symbol of the current node is un complemented
                        then move to the $f$ neighbor and then to the $g$ neighbor
                        else move to the $f$ neighbor
            end
    end
else
    begin
        for $i = 1$ to $n - j_R - m_2$ do
            Move to the $f^{-1}$ neighbor of the current node.
        if $(n - j_R - m_2) \geq 0$ then
            begin
                Move to the $g$ neighbor of the current node.
                for $i = 1$ to $(n - j_R - m_2) + (k - 1)$ do
                    if the first symbol of the current node is un complemented
                        then move to the $f$ neighbor and then to the $g$ neighbor
                        else move to the $f$ neighbor
            end
        else
            begin
                for $i = 1$ to $(k - 1)$ do
                    if the first symbol of the current node is un complemented
                        then move to the $f$ neighbor and then to the $g$ neighbor
                        else move to the $f$ neighbor
            end
        for $i = 1$ to $j_R - k$ do
            Move to the $f$ neighbor of the current node.
        for $i = 1$ to $j_R - k$ do
            if the last symbol of the current node is un complemented
                then move to the $g$ neighbor and then to the $f^{-1}$ 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 : To get the diameter, we must maximize the distance. Left distance and right distance are maximized by putting $m_1 = 0$ and $m_2 = 0$, i.e.,

$$D_L^{max} = \begin{cases} 2(n-1) + c_1 - c_2 \\ 2n + c_1 - c_2 \end{cases} \qquad D_R^{max} = \begin{cases} 2(n-1) + c_2 - c_1 \\ 2n + c_2 - c_1 \end{cases}$$

Now, $m_1 = 0$ and $m_2 = 0 \Rightarrow c_1 + c_2 = n$. Hence to maximize the distance, we must minimize $|c_1 - c_2|$, i.e. $|c_1 - c_2| = 0$ when n is even and $|c_1 - c_2| = 1$ when n is odd. Under these circumstances, maximum possible value for the distance $D = \min(D_L, D_R)$ becomes $2n - 2$ when n is even and $2n - 3$ when n is odd.

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 $D(x) = 2n - 3$, if n is odd and $D(x) = 2n - 2$, if n is even. Thus, the result follows. $\square$

4 Conclusion

We have developed an optimal routing algorithm as well as determined the diameter of the trivalent Cayley graphs. Thus, these results further enhance the utility of those network graphs.

References

[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.

[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.

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

[PN93] D. J. Pritchard and D. A. Nicole. Cube connected Möbius ladders: an inherently deadlock free fixed degree network. *IEEE Transactions on Parallel and Distributed Systems*, 4(1):111–117, January 1993.

[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.

[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.

[VS95] P. Vadapalli and P. K. Srimani. Trivalent Cayley graphs for interconnection networks. *Information Processing Letters*, 54(6):329–335, June 1995.