Processor Interconnection Networks from Cayley Graphs

STEPHEN T. SCHIBELL
and
RICHARD M. STAFFORD

Editor's Note: This paper will appear in a special issue of *Discrete Applied Mathematics* devoted to Electrical Engineering and Graph Theory.

This paper is divided into three sections. In the first section we discuss Cayley graphs and show how they may be used as a tool for the design and analysis of network architectures for parallel computers. In the second section we present our research on the routing problem. This research can be regarded as a first attempt to find general purpose routing algorithms for interconnection networks. In the last section we consider the problem of constructing Cayley graphs that meet specific design parameters.

INTRODUCTION

One of the most important problems facing technology today is the development of scientific supercomputers. Computer science experts believe that future supercomputers will be based on large-scale parallel processing. Such a computer will have a system consisting of many processors and memories. These machines are commonly known as SIMD (single instruction stream – multiple data stream) and MIMD (multiple instruction stream – multiple data stream) machines. The Connection Machine and the Goodyear MPP are examples of the former, while the NCUBE/Ten and the BBN Butterfly represent the latter class of computer. An essential component of such computers is the interconnection network providing communication among the processors and memories of the system.

The advent of very large scale integration (VLSI) makes it possible to put more processors, which are faster and have more memory, on a single chip. Thus, the interconnection networks of future multiprocessor computing systems may be very complex. Indeed, we are seeing this trend today. The Connection Machine developed by Thinking Machines Inc., consists of 2^{16} single-bit processors all working in parallel!

Interconnection networks are often modeled by graphs. The vertices of the graph correspond to processing elements, memory modules, or just switches. The edges correspond to communication lines. If communication is one-way, the graph is directed; otherwise, the graph is undirected. We point out that a model for the Connection Machine is the 12-dimensional binary hypercube, namely Z_2^{12}. The rationale for 2^{12} vertices vs 2^{16} vertices is that there are 2^{12} chips, each chip having 16 processors. Thus, from a communication viewpoint, there are 2^{12} elements.

Here is an incomplete list of graph properties that a good model might possess: simple and efficient routing algorithms, small diameter, high connectivity, and small degree. Also, one would wish the interconnection network to be as efficient as possible. Ideally one wants each processor to send a message and each memory module to receive a message with each "clock tick." One approach to this problem is to design networks with lots of switching nodes connected in such a way as to ensure multiple memory-processor paths. There is also the "lay-out problem," that is the problem of embedding the graph in a 2 or 3 dimensional Euclidean space in a manner that can be realized in hardware. Additionally, it is desirable that the longest wire link be as short as possible since timing problems arise otherwise. Finding graphs that satisfy these conditions can be a formidable task; in fact, the properties of high connectivity and small degree seem to be inversely proportional to each other. Consequently, in a particular application, trade-offs must be made.

Vertex symmetric graphs are especially well suited as models for interconnection networks because these graphs have the property that the graph viewed from any vertex looks the same. Thus, in such networks the same routing algorithm may be used at each processor. Moreover, the symmetry of the graph minimizes congestion, as traffic is distributed uniformly over all vertices. (Note that a random graph would satisfy the second property but not the first.)

At the 1986 SIAM international conference on parallel processing, Sheldon Akers and BalaKrishnan Krishnamurthy suggested using the theory of groups as a tool to construct "good" vertex symmetric interconnection networks. Their main theme was that finite groups provide a rich source of interconnection networks and that group structure provides an algebraic approach to the design problem. Since that time, there has been an explosion of activity directed towards applying group theory to the design of network architectures for supercomputers.

This paper consists of three sections. In the first section we introduce the notation and terminology and provide an exposition of this exciting new field. In the second section we present our research on the routing problem. Routing is the problem of communicating efficiently among the processors and memories. Usually a routing algorithm is network dependent, that is, given a network, one must find a routing algorithm for that specific network. We present in this paper a routing algorithm for any computer architecture satisfying certain properties. Moreover, we demonstrate that our algorithm is extremely efficient in many cases. In the third section we consider the problem of constructing Cayley graphs that meet specific design parameters. In particular, we present research done in support of an effort to study the influence of these parameters on network performance.

1. MATHEMATICAL STRUCTURES FOR COMPUTER NETWORKS

In this section we discuss Cayley graphs and indicate why they may be good models of network architectures for supercomputers. We shall also present an overview of the work of Sheldon B. Akers and BalaKrishnan Krishnamurthy. We assume the reader is familiar with the basic definitions, concepts, and results of graph theory and group theory as found in [5] and [7].

Let G be a group and let Δ be a generating set for G which is closed under inverses. The Cayley graph $\Gamma = \Gamma(G, \Delta)$ is the graph whose vertex set and edge set are

$$V = G \,, \qquad E = \{\{g, h\} \mid hg^{-1} \in \Delta\}.$$

We record some basic facts about Cayley graphs.

Proposition 1.1. Let Δ be a set of generators for a group G. The Cayley graph $\Gamma(G, \Delta)$ has the following properties:

(i) $\Gamma(G,\ \Delta)$ is a connected regular graph of degree equal to the cardinality of Δ;
(ii) $\Gamma(G,\ \Delta)$ is a vertex symmetric graph.

Proof. (i) This follows directly from the definition of a Cayley graph.

(ii) We need to show that the automorphism group of the graph $\Gamma(G, \Delta)$ acts transitively on the vertex set G. For $g \in G$, let ϕ_g be the element of S_G defined by $h\phi_g = hg \; \forall h \in G$. If $\{h, k\} \in E$, then since $(k\phi_g)\,(h\phi_g)^{-1} = kgg^{-1}h^{-1} \in \Delta$, we have $\{h\phi_g, k\phi_g\} \in G$. Thus the elements ϕ_g are permutations of the vertex set G which also preserve the incidence relation of the graph $\Gamma(G, \Delta)$, hence are automorphisms of Γ. Transitivity follows now by noting that for any two elements $g, h \in G$, $g\phi_{g^{-1}h} = h$.

Cayley graphs are actually labeled graphs. The edges are labeled by the elements of Δ. An edge $\{g, h\}$ is labeled by an $x \in \Delta$ with an arrow pointing in the direction of h, i.e.,

$$g \xrightarrow{x} h$$

if and only if $hg^{-1} = x$.

The Alternating group A_4 provides an example to which we refer throughout the paper. The permutations

$$a = (1, 2)\,(3, 4), \text{ and}$$
$$b = (1, 2, 3)$$

generate A_4. Let Δ be the set $\{a, b, b^{-1}\}$. Figure 1 is a picture of the Cayley graph $\Gamma(A_4, \Delta)$.

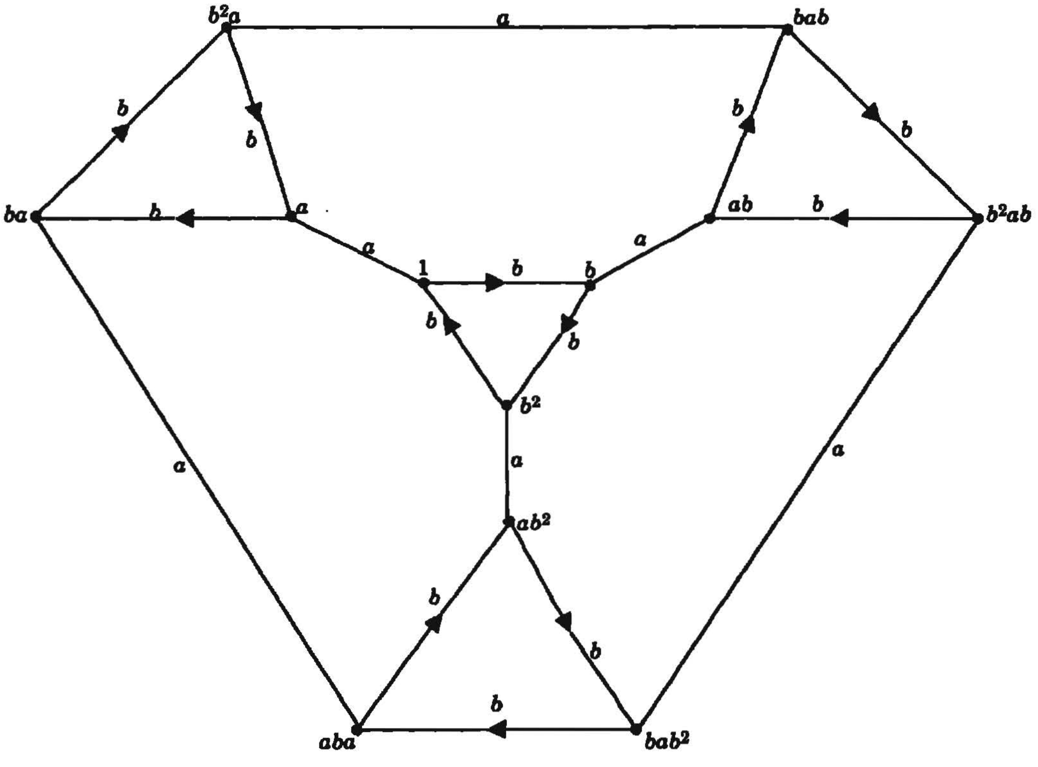
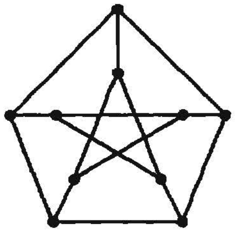


Fig. 1. Cayley graph $\Gamma(A_4, \Delta)$

Notice that this symmetric graph has degree 3. This corresponds to the number of distinct generators, namely a, b, and b^{-1}. Moreover, one can think of the generators as "direction signs". Suppose, for example, one is at the vertex labeled b^2. You may traverse in the direction b to the vertex labeled 1, or you may move in the a direction to the vertex labeled ab^2, or you may move in the direction b^{-1} to the vertex labeled b.

Since $a=a^{-1}$, we have adopted the convention of not assigning an "arrow" to the edge labeled by a. In general, a generator will not be its own inverse as is the case with b. So an edge with an arrow has two labels; it is labeled b in the direction of the arrow and labeled b^{-1} in the opposite direction of the arrow. We suppress the b^{-1} labeling by convention.

We note the following about symmetric graphs. The converse of Proposition 1.1 is false. That is, not all symmetric graphs are Cayley graphs. The simplest counter-example is Petersen's graph below. We leave the proof of our assertion to the interested reader. The Petersen graph is not a planar graph, that is where two edges meet is not necessarily a vertex. We have indicated the vertices by dots.



Petersen's Graph

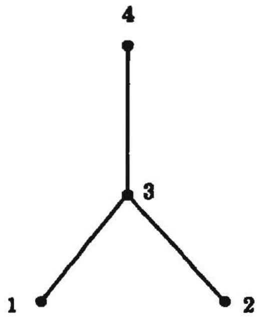
1.1. The Cayley Graph Model

We mentioned in the introduction that vertex symmetric graphs make "good" interconnection networks. Indeed, most of the computers in service today that are based upon large-scale parallel processing have interconnection networks that are vertex symmetric graphs. For example, the Connection Machine has a network architecture that can be modeled by the 12-dimensional binary hypercube. The 256×256 torus-connected 2-dimensional mesh is the architecture of the MPP at the NASA/Goddard Space Flight Center. Finally, the butterfly network and the cube-connected cycle network are also vertex symmetric graphs that are widely accepted as models for network architectures. **Our basic working hypothesis is that network architectures should be vertex symmetric graphs. The central problem then is to find new symmetric graphs that provide superior performance as computer architectures.**

In the previous section we learned how to construct vertex symmetric graphs from groups. That is, if Δ is a generating set for a group G, then by Proposition 1.1, the Cayley graph $\Gamma(G, \Delta)$ is a vertex symmetric graph. Thus, finite groups provide an infinite source of vertex symmetric graphs. In addition, graph theoretic properties are reflected in the algebraic structure of the group and vice versa. Over the past 100 years mathematicians have developed powerful tools with which to study the internal structure of finite groups. Consequently, this vast theory can be used to investigate graph theoretic properties of interconnection networks based upon Cayley graphs.

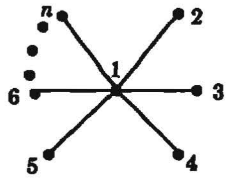
This important observation was made by Sheldon Akers and BalaKrishnan Krishnamurthy in [1]. Using this group theoretic approach, they found two new families of vertex symmetric graphs that they called star graphs and pancake graphs [1]. They also showed that these new interconnection networks in many ways were superior to the n-dimensional binary hypercube and the cube-connected cycle networks.

The star and pancake graphs are Cayley graphs. The vertex set of both of these graphs is the symmetric group on Ω, where $\Omega=\{1, 2, 3, \cdots, n\}$. So all that remains is to define the associated generating sets. To that purpose we need some more definitions. A permutation on Ω is called a transposition provided it interchanges two points and fixes all others. For example, the permutation (3, 4) is a transposition. There is a nice way of representing a set of transpositions pictorially. Namely, we associate with any set of transpositions Δ a unique graph called the transposition graph. The vertices of the graph are labeled with the symbols $\{1, 2, 3, \cdots, n\}$. The edge set, E, is defined by $ij \in E$ if and only if the transposition $(i, j) \in \Delta$. For example, the figure below represents the set of transpositions $\{(1, 3), (2, 3), (3, 4)\}=\Delta$.

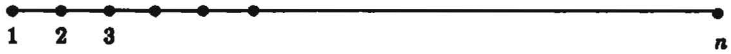


We warn the reader that the above graph is not the Cayley graph determined by Δ, but just a way of pictorially representing the set Δ. The Cayley graph determined by Δ in our example has 24 vertices and is of degree 3.

The transposition graphs that determine the generating set for the star and pancake graphs are



Transposition graph for the star graph



Transposition graph for the pancake graph

S. Akers and B. Krishnamurthy found these networks to be superior to the binary n-cube when measured by their degree, diameter, and connectivity. In fact, they found that star graphs not only possess maximum connectivity but provide minimal degradation of performance in the presence of (a tolerable number of) faults. For a detailed discussion of this see "The Fault Tolerance of Star Graphs" [2]. Table 1 (reproduced directly from [1]) shows that star graphs, when measured solely by degree and diameter, are superior to the binary n-cube.

Table 1. A Comparison

The binary hypercube				The star graph			
n	Size 2^n	Degree n	Diameter n	n	Size n	Degree $n-1$	Diameter $\lfloor(3/2)(n-1)\rfloor$
7	128	7	7	5	120	4	6
8	256	8	8	6	720	5	7
9	512	9	9	6	720	5	7
10	1024	10	10	7	5040	6	9
11	2048	11	11	7	5040	6	9
12	4096	12	12	7	5040	6	9

One obvious drawback of both star and pancake graphs, since their vertex set has cardinality $n!$, is that they are extremely sparse. In fact, there are only nine of each type within a range of three million vertices!

We end this section with a discussion of two design issues that suggest "good" interconnection networks should be large graphs of small degree and small diameter.

The first design issue is to design a network with transmission delays as small as possible. Since the maximum number of links used to transmit any single message is the diameter of the graph, one would think one should make the diameter of the graph as small as possible.

A general rule of thumb for the total cost of a supercomputer is that two thirds of the total cost is due to the processor and memory modules and one third of the cost is the network itself. It is estimated that as much as one third of the network cost is related to the total number of wires; this cost includes the expense of driving messages at very high rates through the wires. Let Γ be an interconnection network with n vertices and e edges. If Γ is a vertex symmetric graph of degree d, one easily computes that

$$e = \frac{nd}{2}.$$

Thus, decreasing the degree of a vertex symmetric graph decreases the total number of wires used to connect the processors, effectively decreasing the total cost. We also mention that it appears that the lay-out problem is easier to solve for low degree networks.

We now present some evidence that Cayley graphs of nonabelian groups and in particular, Cayley graphs of the nonabelian simple groups, may provide the best interconnection networks, at least in the sense of producing graphs of small degree and diameter.

Our first piece of evidence is a result of P. McKenzie; see [9] for details.

Proposition 1.2. Let G be a permutation group on a set Ω of cardinality n. Suppose Δ is a set of permutations that generate G, all of which move at most k points. Then the diameter of $\Gamma(G, \Delta)$ is bounded above by $2(kn)^{2k}$.

L. Babai, W.M. Kantor, and A. Lubotzky, [4], have a result that suggests that the simple groups may be a rich source of large Cayley graphs of small degree and diameter. They prove:

Proposition 1.3. Every nonabelian finite simple group has a set Δ of ≤ 7 generators such that the resulting Cayley graph has diameter on the order of $\log_2|G|$.

This suggests the following conjecture that may be found in [3].

Conjecture. There exists a constant c such that for every nonabelian finite simple group G, the diameter of every Cayley graph of G is bounded above by a number that is on the order of $(\log_2|G|)^c$.

The binary n-cube has size 2^n and diameter $\log_2(2^n) = n$, but its degree is n. The above theorems suggest that the finite simple groups should produce Cayley graphs comparable with the n-cube but of very small degree. In fact, if the conjecture is true, one would expect to find Cayley graphs of these groups with much smaller degree and diameter than the corresponding n-cube of the same size.

2. THE ROUTING PROBLEM

Routing is the problem of communicating efficiently among the processors and memories of an interconnection network. Graph theoretically this problem is equivalent to finding paths between pairs of vertices.

The task of finding paths from one vertex to another in a graph has been extensively studied and there exist many algorithms for this purpose. Dijkstra's algorithm, for example, finds the shortest paths between any pair of vertices. This algorithm can be used in any graph (directed or undirected). The problem with all of these algorithms is that they require an excessive amount of overhead. That is, too much of the computer's resources must be allocated to routing.

The solution at the moment is to design routing algorithms for each specific network. These special purpose algorithms usually only apply to the interconnection network they were intended for. For example, the routing algorithm used in the Connection Machine depends totally on the geometry of the 12-dimensional binary n-cube and is completely different from the routing algorithm used in the MPP.

The main purpose of this section is to present our own research on this problem. **Our research can be regarded as a first attempt to find general purpose routing algorithms for interconnection networks.** Specifically, we present a routing algorithm for any Cayley graph of a permutation group satisfying certain properties. Moreover, we will demonstrate that our algorithm in many cases is extremely efficient. In addition, we shall present some promising new interconnection topologies.

All of the groups we study in this section will be permutation groups. In light of Cayley's theorem we have lost no generality.

2.1. Two Equivalent Problems

In this section we establish the fact that routing in a Cayley graph is equivalent to a special type of factoring in the underlying group.

We first look at an example. Consider the Cayley graph of the permutation group A_4 in figure 2. Suppose one wishes to send a message from the vertex labeled 1 to the vertex labeled bab. There are many different paths that lead from 1 to bab. In figure 2 we have indicated three paths from 1 to bab. From the definition of a Cayley graph and the fact that the vertex labeled 1 is the identity, the path 1 yields $ab^{-1}a = bab$, path 2 yields the obvious factorization of bab, namely bab itself, and path 3 yields $b^{-1}abab^{-1} = bab$. Thus, we have three different factorizations of the element bab. The point is that any path from 1 to bab produces a factorization of bab as a product of elements of the set $\Delta = \{ a, b, b^{-1} \}$.

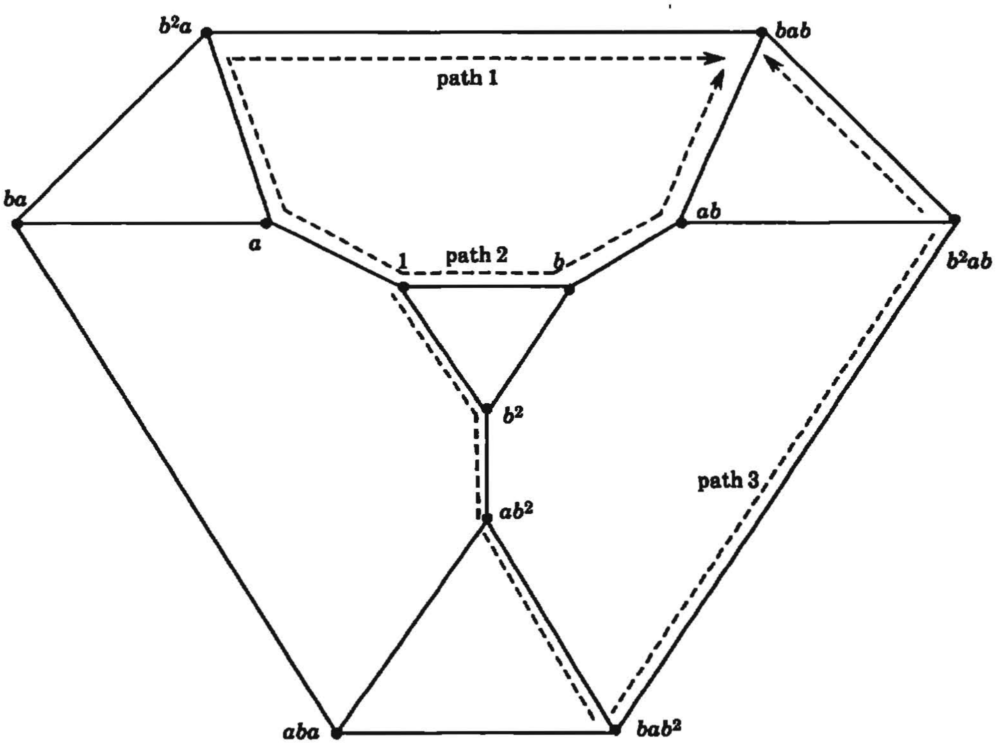


Fig. 2. Cayley graph of the permutation group A_4

The converse of this is also true. Namely, any factorization of bab as a "word" in the generators $\{ a, b, b^{-1} \}$ produces a path from 1 to bab. We record and prove this easy but important fact about Cayley graphs.

Proposition 2.1. Factoring elements in G as "words" in the generators is equivalent to routing in the Cayley graph $\Gamma(G, \Delta)$.

Proof. First suppose we possess an algorithm **A** that can produce a path between any pair of vertices in our Cayley graph $\Gamma(G, \Delta)$. Also suppose that g is an arbitrary element of G. Apply algorithm **A** to produce a path from the identity vertex 1 to the vertex labeled g. Suppose this path is 1, s_1, s_2s_1, $\cdots$, $(s_t \cdots s_2s_1)$. By the definition of a Cayley graph it follows that g is the product $s_t \cdots s_2s_1$, and thus we have factored g as a "word" in the generating set $\{ a, b, b^{-1} \}$. Next assume we have a factoring algorithm F that can express any element g of G as a product of elements a, b, and b^{-1}. Let x and y be two vertices of the Cayley graph $\Gamma(G, \Delta)$ and set $g=yx^{-1}$. Now apply the factoring algorithm F to produce $s_t \cdots s_2 s_1 = g$. Clearly x, $s_1 x$, $(s_2 s_1)x$, $\cdots$,$(s_t \cdots s_2 s_1)x=y$ is a path from x to y.

The problem of factoring in the context of permutation groups has been studied extensively. In fact, if the generating permutations satisfy certain properties then an extremely efficient factoring algorithm does exist. This is the topic of the next section.

2.2. *Factoring in Permutation Groups*

Let Ω be a finite set. Recall that G is said to be a permutation group if G is a subgroup of S_Ω (the symmetric group on Ω). Since G can be very large even when Ω is relatively small, group theorists often describe permutation groups by defining them as the group generated by a set of permutations. In general, for an arbitrary generating set Δ of G, it can be very difficult and computationally prohibitive to determine the order of G or to test an arbitrary permutation for membership in G as well as factoring such a permutation as a word in the generating set Δ. This caused Charles Sims to introduce the fundamental concepts of base and strong generating set [14].

A base for a group $G \subseteq S_\Omega$ is defined to be an ordered subset $B \subseteq \Omega$ with $bg=b$, $\forall b \in B \Rightarrow g=e$, the identity permutation. Heuristically, a base is a large enough subset of Ω that any permutation of G is completely determined by its action on the base. A set of generators Δ of G is said to be a set of strong generators with respect to $B=\{a_1,a_2,\cdots, a_t\}$ provided Δ contains a set of generators for the stabilizing sequence of subgroups G_{a_1}, $G_{a_1 a_2}, \cdots, G_{a_1 \cdots a_t}$. Here $G_{a_1 \cdots a_k}$ is the subgroup $\{g \in G \mid a_i g = a_i, 1 \leq i \leq k\}$.

We remark that our generic example of a Cayley graph (figure 1) provides us a first example. Here the generating set $\Delta=\{a, b, b^{-1}\}$ is a set of strong generators with respect to the base $B=\{4, 1\}$. To see this, one checks that G_4 equals the subgroup generated by b, and $G_{4,1}$ is the identity subgroup. Thus Δ contains a set of generators for the stabilizing sequence G_4, $G_{4,1}$. It is also immediate that the only permutation of A_4 that fixes both 1 and 4 is the identity.

Given a base and strong generating set relative to this base, the above questions are easy to answer. In particular, if the base is small relative to Ω the Sims algorithm is extremely efficient. In the next section we will present a brief description of this algorithm.

2.3. *The Sims Factoring Algorithm*

Let G be a permutation group with strong generators Δ and base B as defined in section 2.2. Also set G^i to be the stabilizer subgroup $G_{a_1 a_2 \cdots a_{i-1}}$,where G^1 is understood to be G.

Proposition 2.2. Let U^i be a complete set of coset representatives of G^{i+1} in G^i. Then every element of G has a unique representation of the form $U_b U_{b-1} \cdots U_1$, $U_i \in U^i$.

Proof. We proceed by induction on the cardinality, b, of the base B. If $B=\{a_1\}$, then $a_1 g=a_1$ implies that g is the identity so $U^1=G$ and there is nothing to show, as $g=g$ is a factorization. So suppose $b > 1$ and $a_1 g=x_j \in \Omega$. Since G is transitive on the orbit that contains a_1 and U^1 is a complete set of coset representatives of G_{a_1} in G, there is a unique coset representative $u_{1j} \in \mathrm{U}^1$ with $a_1 u_{1j} = x_j$. Set $g^2 = gu_{1j}{}^{-1}$, $\Delta^2 = \Delta \cap G^2$ and $B^2 = B - \{a_1\}$. It is immediate from the definitions that G^2 is strongly generated with base B^2 and strong generating set Δ^2. Since $g^2 \in G^2$ and B^2 have cardinality $b-1$, g^2 has a unique representation of the form $U_b U_{b-1} \cdots U_2$ by induction. The result is now immediate.

Since $\Delta \cap G^i$ generates G^i, any coset representative $U_i \in U^i$ can be represented as a "word" in the strong generators $\Delta \cap G^i$. The Sims algorithm factors each group element as a unique product of coset representatives. But these coset representatives, U^i, are chosen such that they have minimal length as words in the strong generators. This forces the Sims coset representatives to satisfy the right Schreier property. That is, if $xy \in U^i$ then $x \in U^i$ (for a discussion of this see [8]).

We now define a family of labeled graphs, Γ_i, $1 \leq i \leq b$, analogous to the transposition graphs of section 1. These graphs will be helpful in understanding the Sims cosets U^i. For each base point a_i, define Γ_i to be the graph whose vertex set V_i is the set $\{ a_i g \in \Omega \mid$ for some $g \in G^i \}$. Set $E_i = \varnothing$, $U^i = \varnothing$, $P =$ identity subgroup and $V^* = \{a_i\}$. We define the edge set E_i inductively as follows:

```
Step i: If $V^* = V_i$ stop
      For each $x \in \Delta \cap G^i$ and each $w \in P$ set $z = wx$,
            If $a_i z \notin V^*$ set $E_i = E_i \cup \{a_i w, a_i wx\}$,
                  Set $P = P - \{w\} \cup \{wx\}$, and set $U^i = U^i \cup \{wx\}$
      End;
Set $i = i + 1$
Go to step $i$.
```

Since G^i acts transitively on the vertex set V_i and $\Delta \cap G^i$ generates G^i, the algorithm terminates with a connected tree Γ_i. The Sims coset representatives are the sets U^i, $1 \leq i \leq b$. The reader will observe that there is a one to one correspondence between U^i and the set of all paths in Γ_i beginning with the base point a_i. This observation allows the cosets to be stored in a very efficient way. To that purpose, define F_i to be an $|\Omega|$-long vector; set the ith component of F_i to be zero and if $x_j \notin V_i$ set the jth coordinate to be negative 1. Next suppose that $x_j \in V_i$ and u_{ij} is the unique coset representative in U^i that maps a_i to x_j, and suppose further that $u_{ij} = ws_k$ where s_k is the kth strong generator, then assign the jth component of F_i to be k. The reader will observe that all the coset representatives of U^i can be recovered from F_i. These vectors are called Schreier vectors. Thus we see the storage requirement for the Sims algorithm is minimal. It must store the strong generators as permutations on Ω. It also must store the Schreier vectors. The number of these vectors is exactly the cardinality of the base B. Thus the memory requirement is $(|\Delta| + |B|)$ integer arrays of dimension $|\Omega|$. An easy calculation shows that the number of lookups needed to factor any permutation in G as a product of strong generators is bounded by

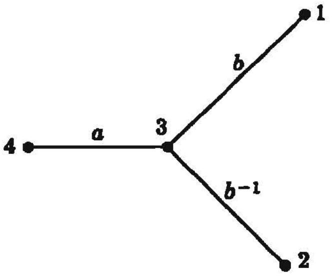
$$|\Omega| \left(\frac{1 + |B| (|\Omega| + 1)}{2} \right).$$

Thus the number of lookups is on the order of $|\Omega|^3$.

Given a permutation $g \in G$ it is a unique product of the form $U_bU_{b-1} \cdots U_1$, $U_i \in U^i$. Each U_i corresponds to a path in Γ_i. This path is the Sims factorization of U_i as a word in the strong generators. The algorithm first examines the image a_1g of the first base point a_1 under g. Since this determines a unique path in Γ_1 from a_1 to a_1g, U_1 is this path. The factorization of U_1 is obtained via the Schreier vector F_1. Now observe $gU_1^{-1} \in G^2$ which is strongly generated by $\Delta \cap G^2$. Thus we proceed inductively to recover U_2 as a word in the generators $\Delta \cap G^2$. We continue in this way to recover each of the U_i in the factorization of g.

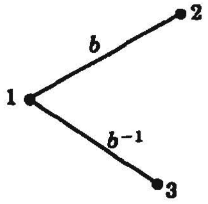
Suppose G is a permutation group on Ω with $|G| = N$, where N is much larger than $|\Omega|$. Suppose further that G has a set of strong generators Δ with respect to some base B. **Then by Proposition 2.1 the Sims factoring algorithm provides an excellent routing algorithm for the Cayley graph Γ(G, Δ). Moreover, we envision each node in the graph to have its own identical algorithm. Thus no global information is needed to route.**

We will illustrate the Sims algorithm with our cannonical example, A_4. Recall in section 2.2 that A_4 has a strong generating set $\Delta = \{a, b, b^{-1}\}$ with respect to the base $B = \{4, 1\}$ where $a = (1, 2)(3, 4)$, $b = (1, 2, 3)$ and $b^{-1} = (1, 3, 2)$. The trees Γ_1 and Γ_2 and the Schreier vectors F_1 and F_2 associated with the base points 4 and 1 appear in figures 3 and 4.



$F_1 = (2, 3, 1, 0)$

Fig. 3. Schrier vector with base point 4



$F_2 = (0, 2, 3 - 1)$

Fig. 4. Schrier vector with base point 1

To illustrate this algorithm we factor the permutation $g = (1, 3, 4) \in A_4$. First note that g moves the base point 4 to the point 1. So we look up position 1 in the Schreier vector F_1 to find generator number 2 which is b. Now we compute the image of 4 under $gb^{-1} = (1, 2)(3, 4)$. Since this is 3, we look at position 3 of F_1 which is generator 1. Next we see that $gb^{-1}a^{-1}$ fixes the base points 4 and 1. Because $\{4, 1\}$ is a base, $gb^{-1}a^{-1}$ is the identity and we have obtained the factorization, namely $g = ab$.

2.4. *Strongly Generated Cayley Graphs*

In this section we shall provide some examples of Cayley graphs whose generators are a set of strong generators for the underlying group. We call such a graph a strongly generated Cayley graph. We remark that by the previous section such graphs have a built-in routing algorithm. But first we obtain an upper bound for the diameter of any Cayley graph that can be given by our representation. Let $\Gamma(G, \Delta)$ be a Cayley graph with G a subgroup of S_{Ω}.

Suppose $|\Omega| = n$, $|\Delta| = m$ and Δ is a set of strong generators for G with respect to some base B with cardinality b. Also let $B = \{a_1, a_2, \cdots, a_b\}$, $G^i = G_{a_1 a_2 \cdots a_{i-1}}$, and n_i be the cardinality of the set U^i. Then we have

Proposition 2.1. The diameter of $\Gamma(G, \Delta)$ is bounded by

$$\sum_{i=1}^{b} (n_i - 1).$$

Proof. Any $g \in G$ can be written as a unique product $U_b U_{b-1} \cdots U_1$, where U_i is a coset representative of G_{i+1} in G_i. It suffices to show that U_i is the product of at most $(n_i - 1)$ members of Δ. Now each $u \in U^i$ has a minimal representation as the product of say $l(u)$ members of $\Delta \cap G^i$. So we define the length of u to be $l(u)$ and set $L = \max \{l(u) \mid u \in U^i\}$. Next pick $u^* \in U^i$ with the length of u^* equal L. Then by the right Schreier property, U^i must have at least L coset representatives of length at least one. Consequently, U^i must have cardinality at least $L + 1$. Since the cardinality of U^i is n_i, the theorem follows.

We introduce a new definition. We define the algorithmic diameter of any Cayley graph $\Gamma(G, \Delta)$ that can be represented by our methods to be the length of the longest factorization given by the Sims algorithm. We remark that our definition may be base dependent.

Example 2.1. The Star graph

In section 1.1 we found that the star graph networks discovered by Sheldon Akers and BalaKrishnan Krishnamurthy had many desirable properties as models for interconnection networks. The reader can check from the transposition graph defining the generating set for the star graph in section 1.1 that $\Delta = \{ (1, 2), (1, 3), (1, 4), \cdots, (1,n) \}$ is the generating set for the underlying group. If one lets $B = \{2, 3, 4, \cdots, n\}$ it is easy to check that Δ is a set of strong generators. Thus the star graph is a strongly generated Cayley graph and consequently our algorithm may be used to route in this family of networks. The authors in [1] calculate the diameter of this family to be

$$\left\lfloor \frac{3}{2}(n-1) \right\rfloor.$$

It would be of interest to compare this with the algorithmic diameter.

Proposition 2.2. The algorithmic diameter of the star graph is bounded above by $2n - 3$.

Proof. Let G be the underlying group of the star graph on n points. Define $G^i = G_{2,3,\cdots,i}$, $i \geq 2$, (that is G^i is the point stabilizer of the points 2 through i) and set G^1 to be G itself. Also let U^i denote the Sims coset representatives of G^i in G^{i-1}. Since G^i is isomorphic to the symmetric group on $n - i + 1$ letters, it follows that U^i consists of $n - i + 2$ cosets. The permutation $(1, i)(1, t)$, $t \geq (i + 1)$ maps the point i to the point t. Thus these $n - i$ permutations are distinct coset representatives of U^i and have length at most 2. Since the permutation (1, i) and the identity are both members of U^i, it follows that all members of U^i have length at most 2. In the case when $i = n$ there is exactly one coset representative namely (1, n). So the algorithmic bound is

$$\leq \left(\sum_{i=1}^{n-2} 2 \right) + 1 = 2n - 3.$$

Example 2.2. The Pancake graphs

The pancake graphs defined in section 1 are strongly generated Cayley graphs. We leave it as an exercise to the reader to check that the set { (1, 2), (2, 3), (3, 4), $\cdots$, ($n-1$, n) } is a set of strong generators with respect to the base {1, 2, 3, $\cdots$, $n-1$}.

Example 2.3. The Mathieu group M_{11}

The sporadic simple group M_{11} of order 7920 has a permutation representation of degree 12. It can be shown that the set $\Delta = \{a_1, a_2, \cdots, a_8\}$ (see table 3 for a definition of these permutations) is a set of strong generators for this group on the base {1, 2, 3, 4}. Also see table 2 for a list of the four Schreier vectors. A calculation shows that the Cayley graph $\Gamma(M_{11}, \Delta)$ has diameter 7, average diameter 5.25, algorithmic diameter 12, and average algorithmic diameter 7.2. Thus this graph of size 7920 has degree 8 and diameter 7. This compares very well with the corresponding hypercube of the same degree that has diameter 8, and 256 vertices!

We next demonstrate our routing algorithm. A computation shows that the permutations $x = (1, 12, 11)(2, 7, 3, 6, 4, 5)(9, 10)$ and $y = (2, 8, 11, 4, 12, 5, 7, 9, 3, 10, 6)$ are elements of M_{11}. We desire to calculate a path from x to y. From Proposition 2.1, we see that we need only factor $yx^{-1} = (1, 11, 6, 5, 2, 8, 12, 4)(3, 9, 7, 10)$ as a word in the strong generators. Note that this permutation moves the first base point 1 to 11. As in our example, we look at position 11 in F_1 which is generator 2. Thus we proceed from x in the "direction" of a_2 to the vertex a_2x. We now need to calculate a path from a_2x to y. But this is equivalent to factoring $yx^{-1}a_2^{-1}$. We proceed inductively. The algorithm terminates when we have to factor the identity element. At this point, we have factored yx^{-1} and have generated a path from x to y.

Table 2. The Schreier vectors for M_{11}

$F_1 = (0, 3, 1, 8, 8, 8, 5, 7, 6, 7, 2, 4)$
$F_2 = (-1, 0, 6, 7, 3, 1, 4, 5, 4, 7, 7, 7)$
$F_3 = (-1, -1, 0, 4, 7, 6, 4, 8, 8, 6, 7, 8)$
$F_4 = (-1, -1, -1, 0, 7, 7, -1, 8, -1, -1, 7, 8)$

The reader will note that in our example the algorithm uniquely factored yx^{-1} as a product of the generators, thus producing a unique path from x to y. This is always the case. In fact, given any group element x, routing from x defines a spanning tree rooted at x. The spanning tree rooted at the identity for our cannonical example A_4 appears in figure 5.

Table 3. Strong generators for M_{11}

$a_1 = (2,6)(3,5)(4,7)(9,10)$
$a_2 = (1,11)(3,5)(2,7)(4,6)$
$a_3 = (2,5)(3,6)(4,7)(11,12)$
$a_4 = (3,4)(7,6)(8,9)(11,12)$
$a_5 = (2,8)(4,9)(5,6)(11,7)$
$a_6 = (8,5)(3,6)(4,10)(11,9)$
$a_7 = (8,11)(4,6)(10,7)(5,12)$
$a_8 = (11,5)(12,6)(4,8)(9,10)$

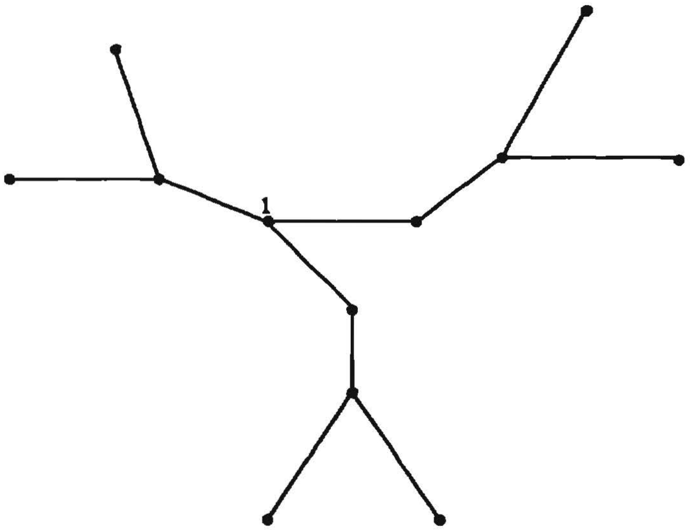


Fig. 5. Spanning tree for Example A_4

2.5. *An "Alternate Path" Algorithm*

In the previous section we computed a path between two elements of the group M_{11}. Recall that at each step we computed which generator should be applied. That is, we traverse the edge labeled by this generator in the Cayley graph. It may happen that we will be unable to traverse this edge due to network loading. For this reason, a simple rule for choosing an alternate next edge, thus an alternate path, is desirable. Our idea is to modify our algorithm to produce alternate paths.

The present algorithm routes on a spanning tree of the Cayley graph. This spanning tree is uniquely determined by the given ordered base for which the generators are strong generators. If there were another base for these generators, then the algorithm implemented with respect to this base would route on a different spanning tree, hence producing alternate paths. Thus our idea is to find generators that are strong with respect to many bases. We could then switch between spanning trees when necessary. Usually this is not possible. However, if we have the luxury of increasing the number of generators (thus increasing the degree of the Cayley graph) it can be accomplished. We illustrate by referring to A_4 again. Figure 6 is the new Cayley graph obtained by adding the generators $c=(2, 4, 3)$ and $c^{-1}=(2, 3, 4)$ to the original generating set for A_4. This expanded generating set is a strong generating set with respect to the ordered base {1, 2}. The spanning tree determined by this new choice of base and strong generators is shown in figure 7. Notice that this tree and that of figure 5 have only three edges in common. In this way we have constructed alternate spanning trees for many strongly generated Cayley graphs, including the Cayley graph of M_{11} presented in this paper. The reader should also observe that the generating sets for the star graphs, respectively the pancake graphs, (see section 1) are strong generators with respect to $(n-1)!$ bases.

The main conclusion of this section is that permutation groups represented by a set of strong generators produce Cayley graphs with an automatic routing algorithm built in, namely the Sims factoring algorithm.

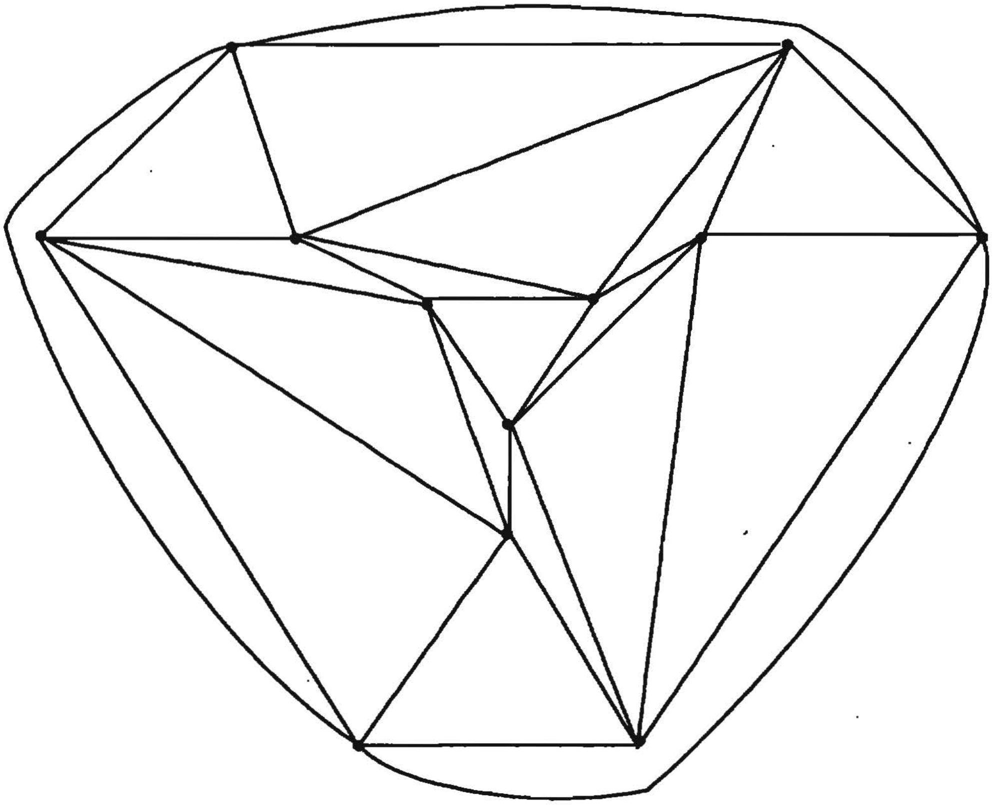


Fig. 6. New Cayley graph obtained by adding the generators $c = (2, 4, 3)$ and $c^{-1} = (2, 3, 4)$ to the original generating set for A_4

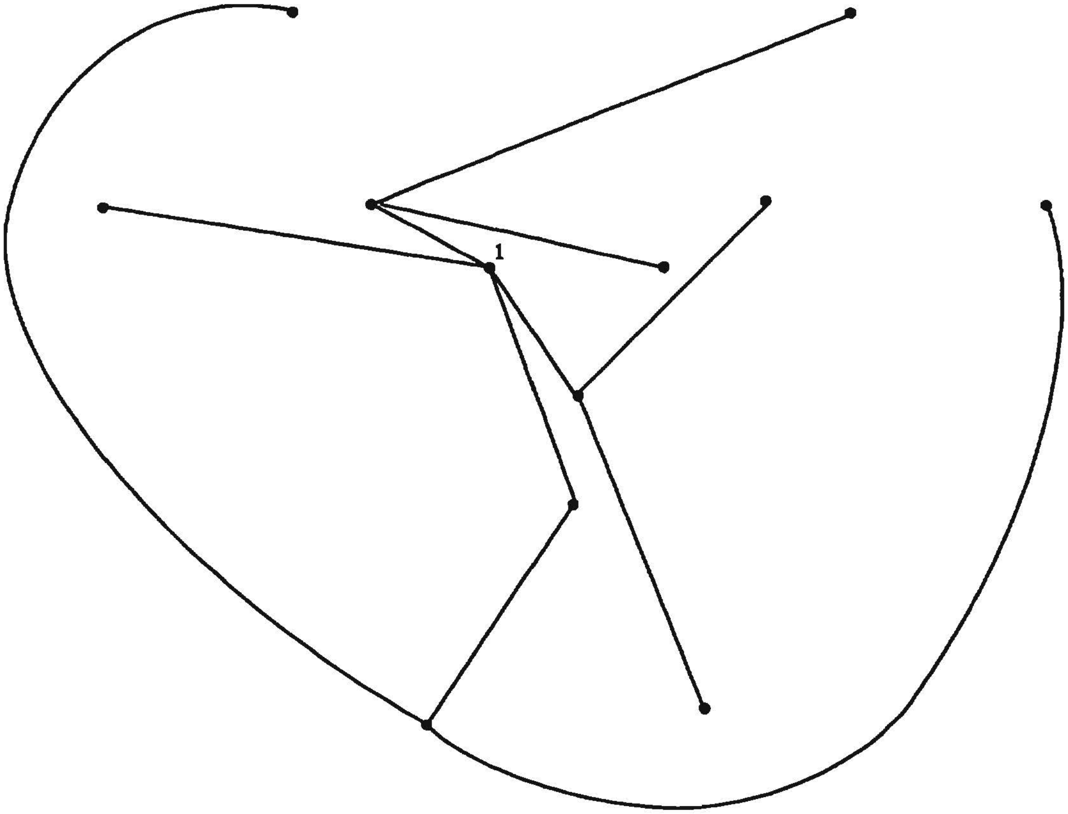


Fig. 7. Spanning tree obtained by adding the generators $c = (2,4,3)$ and $c^{-1} = (2,3,4)$ to the original generating set for A_4

3. DESIGNING "OPTIMAL" NETWORKS

It is believed that the degree of an interconnection network for a large scale, shared memory high performance MIMD machine must be small. For example, Pittelli and Smitley of SRC feel that due to limitations in present day technology, it is not possible to build any of the networks that they have studied if the degree exceeds 6 [11]. Thus in a search for Cayley graph models we must look for groups that are generated by only a few elements. Given that the degree of a network is fixed, it is conjectured that the average diameter is the predominant factor in determining the network performance [11].

Indeed, a recent study by Pittelli and Smitley provides experimental evidence of this [12, 13]. In this section we discuss our contribution to this study. Specifically we were asked to design Cayley graphs to be used in their simmulation. To study the innate performance characteristics of these graphs, it was decided that they would be evaluated at an artificially high 100% message injection rate, and also every node would be a processor or a memory module. Due to real world constraints it was decided that the graphs should have approximately 1024 vertices, be of degree ≤ 6, and have an average diameter ≤ 7.5. The importance of average diameter in determining network performance was supported by the fact that the graphs found by us had the smallest average diameter and out performed all other graphs evaluated in the study. Table 4 lists the graphs evaluated in the study except for the degree 10 binary hypercube that has been included for comparative purposes. The first five graphs are popular parallel processor networks while the last three are our constructions. We will return to this table momentarily.

Table 4

Graph	Vertices	Degree	Diameter	Avg. Diameter
Hypercube	1024	10	10	5.0
Toroid (32 × 32)	1024	4	32	16.0
Toroid (8×8×16)	1024	6	16	8.0
Butterfly (128 × 8)	1024	4	10	6.6
Super Toroid	1024	4	12	6.8
SS1 PSL(2, 13)	1092	4	9	6.2
SS2 Subgp. of M_{24}	1024	5	8	5.2
SS3 Subgp. of S_{16}	1024	6	7	4.5

The nature of this work was experimental as well as theoretical. We would use group theoretic insight to construct candidate Cayley graphs with the appropriate size and degree. We would then calculate the average diameter of the graph. The software package CAYLEY, developed at the University of Sydney, greatly enhanced our ability to examine many Cayley graphs.

Heuristically speaking, since we want to construct graphs with low average diameter we require the generators to have as few "short" relations as possible. The general idea is that if we pick an initial point in the Cayley graph $\Gamma(G, \Delta)$, applying the generators to this point will give us deg (Γ) new vertices in the graph. We repeat the process for each of the new points found except that now, due to relations of the form aa^{-1}, we can pick up at most deg $(\Gamma)-1$ new vertices with each application. Whenever application of a generator branches back to a previously "found" point, it is due to some relation on the generators. Low average diameter graphs should have very little of this branching back phenomenon occurring in the early stages of the process. Hence the Cayley graph should look locally like a tree everywhere. Clearly, abelian groups can not fit this description.

We remind the reader that since our Cayley graphs are undirected, the generating set Δ, defining the graph must be closed under inversion. Thus if $x \in \Delta$, x^{-1} does also. To keep the degree of the Cayley graphs low, we tried to pick generating sets that consisted entirely of involutions, i.e. generators that were their own inverses ($x = x^{-1}$). This seemed to be a good idea and in fact we found that of all our constructions, the Cayley graphs with the lowest average diameters had generating sets satisfying this property.

At the end of section 1 we presented some evidence (Propositions 1.2 and 1.3) that suggested that simple groups "may provide the best interconnection networks, at least in the sense of small degree and diameter." In example 2.3 we saw that the Mathieu group M_{11} with its average diameter of 5.25, is a prime example supporting this suggestion. While simple groups do seem to have desirable average diameters, the sparse distribution of the orders of the simple groups makes it unlikely that there will be many of these suitable for use as realistic interconnection network models. Indeed, PSL(2, 13) is the only simple group appearing in table 4. To overcome this difficulty we looked elsewhere for another source of suitable groups. We did not have to look far. Recently O'Brien has shown that there are 56,092 groups of order 256 [10]. The number of groups of order 1024 is unknown but is probably in the millions, thus a plethora of potential Cayley graphs of the required size awaited our investigation. Since abelian groups have nilpotence class 1, our first intuition was to construct graphs from maximal nilpotence class groups; however, we soon found that we could construct graphs of superior average diameter from groups of lesser class such as a Sylow-2 subgroup of the Mathieu group M_{24} (SS2). This was also the case for our other graphs that were constructed from a subgroup of the Sylow-2 subgroup of S_{16} (SS3). We also point out that maximal nilpotence class groups seem to require a large number of generators, thus increasing the degree of the graph.

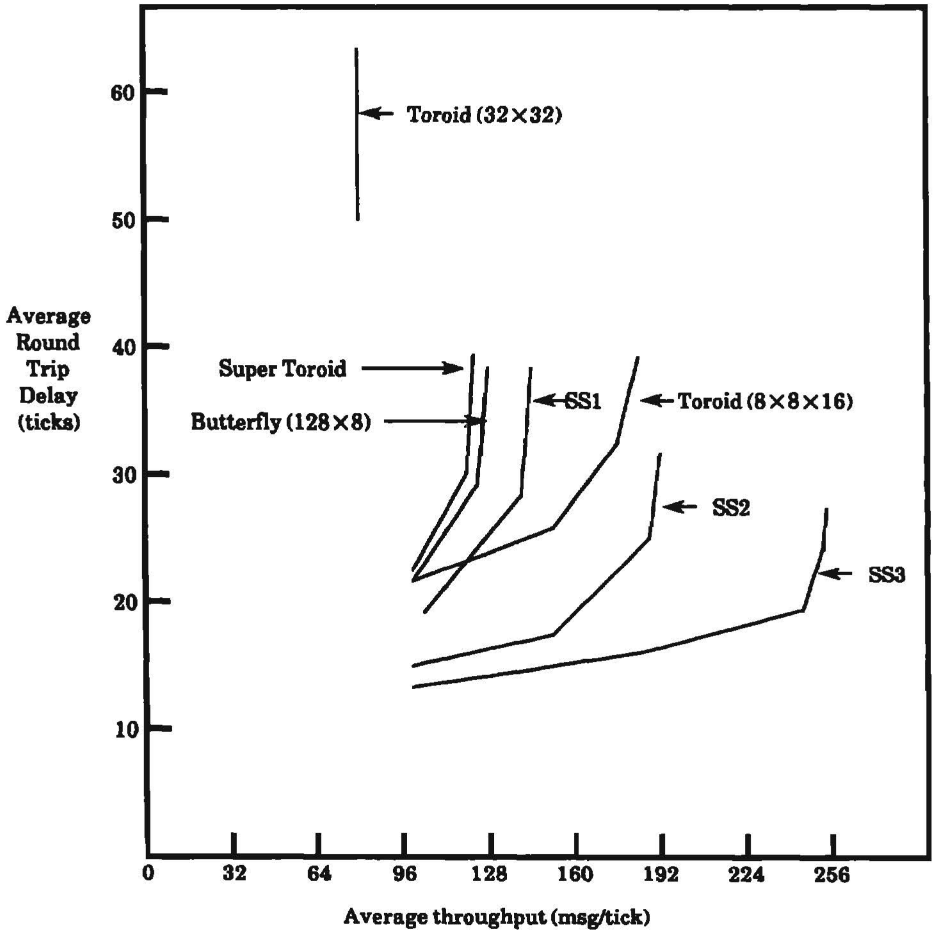


Fig. 8. Performance without queues

Finally we present (courtesy of Pittelli and Smitley) the experimental results alluded to earlier. The reader should consult references [12] and [13] for the specific details of the assumptions and optimizations underlying their network model. Performance is measured in terms of average round trip delay (the number of clock ticks for a message to travel from a processor to a memory module and back) versus average throughput (the average number of messages entering or leaving the network at any instant of time). Figure 8 is the performance plot obtained when the switch nodes have no link queues so that performance is more directly related to the properties of the graph defining the network. The performance gained by adding link queues can be seen in figure 9. In any case the reader should note that PSL(2, 13) outperformed the other degree 4 graphs by a statistically significant margin, as was the case for our degree 5 and 6 graphs also. In fact, before being driven into saturation, PSL(2, 13) sustained 12.5% more network traffic than the next best candidate, a butterfly architecture, and 75% better than the bench mark 2-d mesh.

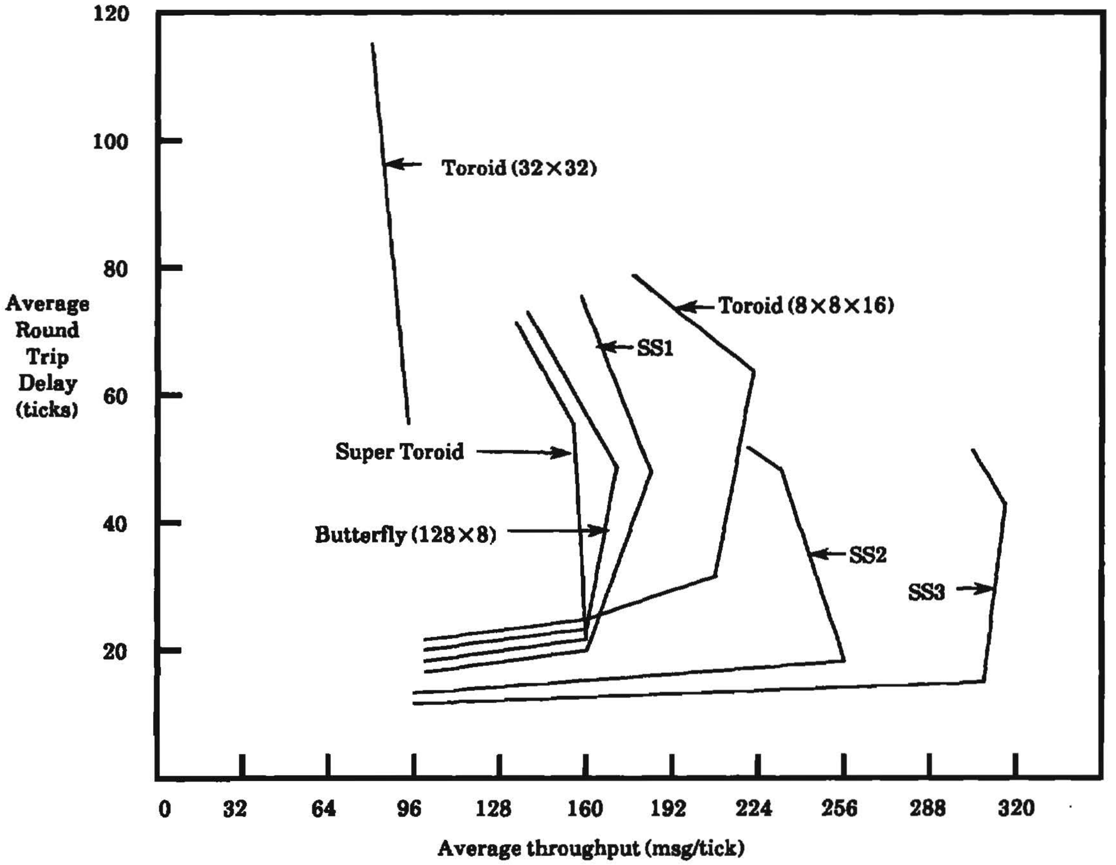


Fig. 9. Performance with queues

STATUTORILY EXEMPT

Acknowledgments

We would like to thank Drs. Frank Pittelli and David Smitley of the Supercomputing Research Center (SRC) for providing insight into hardware issues. We also acknowledge our colleague and friend Dr. Robert Morris for his encouragement and assistance through the research phase of this work as well as during the preparation of this paper.

[redacted]

REFERENCES

[1] Akers, S.B. and B. Krishnamurthy. "A Group Theoretic Model for Symmetric Interconnection Networks." Proceedings of the International Conference on Parallel Processing, pp. 216–23, 1986.

[2] Akers, S.B. and B. Krishnamurthy. "The Fault Tolerance of Star Graphs." Second International Conference on Supercomputing, 1987.

[3] Babai, L. "On the Diameter of Cayley Graphs of the Symmetric Group." *Journal of Combinatorial Theory*. Series A. 49.

[4] Babai, L., W.M. Kantor, and A. Lubotzky. "Small Diameter Cayley Graphs for Finite Simple Groups," submitted.

[5] Bondy, J.A. and U.S.R. Murty. *Graph Theory with Applications*. London: MacMillan, 1976.

[6] Cannon, J. "A Computational Toolkit for Finite Permutation Groups." Proceedings of the Rutgers Group Theory Year 1983–84.

[7] Hall, M. *The Theory of Groups*. New York: Macmillan, 1959.

[8] Leon, J.S. "On an Algorithm for Finding a Base and Strong Generating Set for a Group Given by Generating Permutations." *Mathematics of Computation*, 35, 1980, pp. 941–74.

[9] McKenzie, P. "Permutations of Bounded Degree Generate Groups of Polynomial Diameter." Inf. Proc. Letters, 19, 1984, pp. 253–54.

[10] O'Brien, E.A. Phd. Thesis.

[11] Pittelli, F. and D. Smitley. Personal communication.

[12] Pittelli, F. and D. Smitley. "Analysis of a 3D Toroidal Network for a Shared Memory Architecture." Proceedings of Supercomputing 88 Conference, 1988.

[13] Pittelli, F. and D. Smitley. Unpublished notes.

[14] Sims, Charles C. "Computation with Permutation Groups." Proceedings of the 2nd Symposium on Symbolic and Algebraic Manipulation, 1971.