Dense Gaussian Networks: Suitable Topologies for On-Chip Multiprocessors

Carmen Martínez[1], Enrique Vallejo[1], Ramón Beivide [1], Cruz Izu [2] and Miquel Moretó [3].

Keywords: On-Chip Networks, Topology, Lay-out, Routing, Broadcasting, Circulant Graphs.

Abstract

This paper explores the suitability of dense circulant graphs of degree four for the design of on-chip interconnection networks. Networks based on these graphs reduce the Torus diameter in a factor $\frac{1}{\sqrt{2}}$ which translates into significant performance gains for unicast traffic. In addition, they are clearly superior to Tori when managing collective communications.

This paper introduces a new two-dimensional node's labeling of the networks explored which simplifies their analysis and exploitation. In particular, it provides simple and optimal solutions to two important architectural issues: routing and broadcasting. Other implementation issues such as network folding and scalability by using hierarchical networks are also explored in this work.

[1]Depto. Electrónica y Computadores. Universidad de Cantabria. Avenida de los Castros s/n 39005 Santander, Spain. Email: carmen.martinez@unican.es, {enrique, mon}@atc.unican.es.

[2]Dept. Of Computer Science. The University of Adelaide. Adelaide, SA 5005, Australia. Email: cruz@cs.adelaide.edu.au.

[3]Depto. Arquitectura de Computadors. Universitat Politcnica de Catalunya. Campus Nord, 08034 Barcelona, Spain. Email: mmoreto@ac.upc.edu.

1 Introduction

The increasing number of transistors per chip has lead to the design of multiprocessors in a single die. These chips containing multiple processor cores are denoted as on-chip multiprocessors (CMPs). Large scale systems, such as Piranha [1] and IBM Power4 [17], combine multiple CMPs to obtain higher performance. The problem of choosing the appropriate architecture for implementing a CMP is still open nowadays. One proposed solution is to employ point-to-point on-chip networks [10]. In this way, the resulting regular wiring scheme allows to reuse highly optimized system components, including wiring layouts.

With current technology, on-chip networks have to be arranged in two dimensions. We consider in this paper suitable 2D topologies for the design of on-chip networks which minimize distances among nodes. We will model the network by means of its associated graph, processors being represented as graph nodes and communication links as the edges connecting them. Two basic distance-related graph parameters are diameter and average distance. Both diameter (longest path among nodes) and average distance should be as low as possible to minimize communication delays.

The simplest bi-dimensional topology is a 2D Mesh, whose longest path connects any pair of nodes located in opposite corners. Thus, the diameter of a N-node 2D Mesh is $2(\sqrt{N}-1)$. The torus adds $2\sqrt{N}$ wrap-around links to an $N-node$ Mesh, which reduces its diameter to $\sqrt{N}$ or $\sqrt{N}-1$ depending on whether $\sqrt{N}$ is even or odd. It also provides symmetry, a very desirable topological property because it simplifies network analysis and design.

In real life, we reduce the distance between two points in a plane, by traveling via the shortest (Euclidean) path. If the same were possible for messages traveling among nodes in a square lattice, their longest path would be the distance between the farthest nodes

(the diagonal), that is $\sqrt{2}\sqrt{N}$.

We will show that it is possible to find a mesh-like graph that halves this maximum distance by adequately connecting its wrap-around links. Its diameter is $\left\lfloor \frac{\sqrt{N}}{\sqrt{2}} \right\rfloor$. Hence, messages traveling between the farthest nodes will use paths whose distances are bounded by half of the maximum Euclidean distance in a square of size $\sqrt{N}\sqrt{N}$. These networks can be successfully applied to the design of on-chip parallel systems. As we will see, for the same number of nodes and links, their richer connectivity and lower diameter make them topologically superior to Tori for both individual and collective communications.

The networks presented in this paper are based on the family of dense degree four Circulant graphs, that is, containing the maximum number of nodes for a given diameter. Circulant graphs have been used for decades in the design of computer and telecommunication networks due to their optimal fault-tolerance characteristics and their simple routing algorithms [6]. The name *Circulant* comes from the nature of its adjacency matrix; a matrix is circulant if all its rows are periodic rotations of the first one. The family of circulant graphs includes among its members the Complete graph and the Cyclic graph (Ring).

Traditionally, the N nodes of a circulant graph have been labeled by means of the subset of integers ranging from zero to $N-1$. A previous paper has shown that Gaussian integers, or the subset of the complex numbers with both real and imaginary integer parts, provide the appropriate mathematical model to deal with a subfamily of circulant graphs denoted as Gaussian graphs [12]. As these networks are based on Gaussian graphs containing the maximum number of nodes for a given diameter, we denote them as *Dense Gaussian Networks* (DGNs). One of the advantages from considering Gaussian integers to model these graphs is the existence of an adequate two-dimensional labeling of their nodes. There are many applications over dense Gaussian networks that can benefit from this new bi-dimensional labeling. We will consider in this paper some of them such as

unicast and broadcast packet routing which lead to simple hardware implementations and the design of hierarchical networks.

The rest of this paper is organized as follows. Section 2 motivates the suitability of DGNs for on-chip networks by comparing them versus Tori topologies. Next, Dense Gaussian Networks are defined in Section 3. Section 4 presents an optimal routing algorithm for DGNs which only uses sums and comparisons. Section 5 describes a broadcast algorithm based on a geometrical interpretation of DGNs. Section 6 considers implementation issues for DGNs such as two-dimensional folding and hierarchical based topologies. Finally, Section 7 concludes the main achievements of the paper.

2 Motivation

This Section motivates the suitability of dense Gaussian networks for the design of on-chip networks. We present the main characteristics of these networks in comparison to Tori. Both networks are degree four symmetric graphs containing N nodes and connecting them by the same number of links, $2N$. However, as we will see later in the paper, the diameter of the dense Gaussian network is just around 70% of the diameter of a Torus of the same size. The richer connectivity of a DGN has as a counterpart a higher number of wrap-around links in its two-dimensional layout. The difference between the number of wrap-around links used by both networks is around a 6%. Then, a 30% diameter reduction is achieved by employing only about 6% more wraparound links. This seems to be a manageable cost when considering the impact that the diameter has on network performance.

It has been previously proved that reducing topological distances by skewing the wrap-around links in rectangular and "L-shape" Tori, results in better system performance [7], [15], [20]. Having lower distances implies higher network throughput and lower packet latencies which reduce the execution times of typical applications running over different

kinds of multiprocessor platforms.

Nevertheless, we need to compare the two networks not only from the topological point of view, in which the dense Gaussian network is the clear winner, but also in terms of their cost, performance and implementation. Consequently, we will devote the rest of this Section to explore different factors that make a network topology suitable for an on-chip parallel system. For each one, we will describe how the DGN fares against the Torus.

It is clear that networks should be deadlock-free and provide adaptive minimal routing at a reasonable cost even in the presence of failures. A minimal and easy to implement unicast packet routing will be considered in Section 4. Based on this new mechanism, simple and efficient adaptive routing and deadlock-avoidance mechanisms defined for Tori can be easily exported to dense Gaussian networks. For example, the adaptive Bubble routing algorithm proposed in [16] for Tori can be successfully used in Gaussian networks, even in the presence of arbitrary failures. The implementation costs are identical for both topologies. That routing mechanism has one of the best cost/performance ratio, and has been applied to the design of the torus network for the IBM BlueGene/L supercomputer [5].

An optimal network should efficiently support collective communications like one-to-all and all-to-all broadcasting and reductions. Modern cache coherency protocols [11] and synchronizing barriers implementation are based on broadcast trees. As the connectivity pattern of dense Gaussian networks allows to reach the maximum number of nodes for a given diameter, a broadcast tree can be traversed in the shortest possible time. We present in Section 5 a broadcast algorithm that is universal for every node and ends in time proportional to the diameter of the network, which is $\sqrt{\frac{N}{2}}$. A similar broadcast in a Torus needs $\sqrt{N}$ steps. In addition, a hardware implementation of our one-to-all broadcast is much simpler than equivalent optimal mechanisms for Torus networks [21]. Reduction collective operations can also beneficiate from DGN connectivity as they employ a similar

communication topology.

Finally, the network should be easily implementable on a VLSI chip. The number and shape of wrap-around network links has a significant impact on its final layout. Minimizing the number of wire's crosses and equalizing their lengths should be goals to be pursued in order to achieve a scalable network design. The wrap-around connectivity of the dense Gaussian network makes it difficult to produce such a compact layout. Nevertheless, as we will see in Section 6, a folding technique can be applied over Gaussian networks for obtaining a lay-out similar to the one employed by folded Tori. We provide a method in which physical distances among nodes are equalized and the maximum wire length for the resulting layout is $\sqrt{5}$, regardless the number of network nodes. Furthermore, no more than four metal layers are needed for a complete planar implementation.

3 Dense Gaussian Networks

As mentioned before, dense Gaussian networks are built over circulant graphs. The vertex-symmetry of circulants allows their analysis starting from any vertex (node zero unless any other is stated), which simplifies their study. By exploiting this property, degree four circulants have traditionally been studied by means of plane tessellations, [8], [19].

A *Circulant graph* with N vertices and jumps $\{j_1, j_2, \ldots, j_m\}$ is an undirected graph in which each vertex n, $0 \leq n \leq N-1$, is adjacent to all the vertices $n \pm j_i$, with $1 \leq i \leq m$. We denote this graph as $C_N(j_1, j_2, \ldots, j_m)$. It is clear that a circulant graph $C_N(j_1, j_2, \ldots, j_m)$ is connected if and only if $\gcd(j_1, j_2, \ldots, j_m, N) = 1$. In this case, the circulant $C_N(j_1, j_2, \ldots, j_m)$ is a regular graph of degree $2m$ since every vertex is connected to exactly $2m$ vertices. Figure 1 shows the degree four circulant graph $C_{25}(3, 4)$.

In a degree four circulant graph there can be, at most, $4d$ different nodes at distance d from node 0. Thus, for a given diameter k the maximum number of nodes of a $C_N(j_1, j_2)$

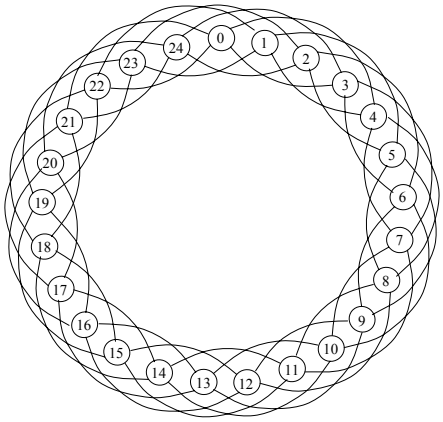


Figure 1: $C_{25}(3,4)$ Circulant Graph.

graph is:

$$N = 1 + 4\sum_{d=1}^{k} d = 1 + 4\left(\frac{k(k+1)}{2}\right) = 2k^2 + 2k + 1.$$

Graphs containing such a maximum number of nodes can be denoted as *dense* degree four circulants. Different authors have shown that $C_N(k, k+1)$ graphs with $N = 2k^2 + 2k + 1$ are dense degree four circulants, [3], [4], [6], [2]. The circulant depicted in Figure 1 is, actually, one of these dense graphs for $k = 3$ and $N = 25$. It is easy to infer from the previous expression that the diameter of a $C_N(k, k+1)$ graph is $k = \left\lfloor \sqrt{\frac{N}{2}} \right\rfloor$. In the same way, as there are $4d$ different nodes at distance d from node 0, the average distance of a $C_N(k, k+1)$ graph is:

$$\bar{k} = \frac{4\sum_{d=1}^{k} d^2}{N-1} = \frac{4k(k+1)(2k+1)}{6(2k^2+2k)} = \frac{2k+1}{3} \cong \frac{2}{3}\sqrt{\frac{N}{2}}.$$

Gaussian networks are based on these circulant graphs of degree four but they employ a two-dimensional labeling of their nodes which facilitates their analysis and exploitation.

With this new labeling nodes are represented with two integer coordinates. Next, we define Dense Gaussian Networks.

Definition 1 *Let k be a positive integer. The Dense Gaussian Network of diameter k, or G_k, is defined as follows:*

- *The square $Q_k = \{(x, y) \in \mathbb{Z} \times \mathbb{Z} \mid |x| + |y| \leq k\}$ is the set of nodes and*
- *Every node $(x, y) \in Q_k$ is adjacent to the nodes $(x + 1, y)$, $(x - 1, y)$, $(x, y + 1)$ and $(x, y - 1)$* `MOD` *$(k, k + 1)$,*

where the equivalence relation `MOD` *is defined as follows:*

$$(x, y) \equiv (x', y') \texttt{ MOD } (k, k+1) \Leftrightarrow \exists u, v \in \mathbb{Z} | (x = x' + uk - v(k+1)) \wedge (y = y' + u(k+1) + vk).$$

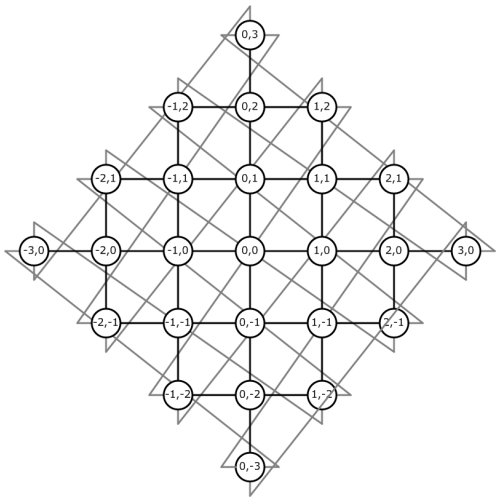


Figure 2: Dense Gaussian Network G_3.

We can see in Figure 2 the circulant graph $C_{25}(3, 4)$ of Figure 1 as a dense Gaussian network with diameter $k = 3$. As an example of how `MOD` function works, consider node $(1, 2) \in Q_3$. This node is adjacent to nodes $(0, 2)$ and $(1, 1)$ inside the mesh. The wrap-around links that connects the node $(1, 2)$ to its other two adjacent nodes are determined by:

- $(1+1,2) = (2,2) \equiv (-1-2)$ MOD $(3,4)$ taking $u = 1$, $v = 0$.
- $(1,2+1) = (1,3) \equiv (-2,-1)$ MOD $(3,4)$ taking $u = 1$, $v = 0$.

Note that this modulo function is only necessary for determining peripheral adjacency among nodes. Actually, this two-dimensional modulo function is the modulo reduction over the complex numbers with real and imaginary integer parts or Gaussian integers. Moreover, this modulo operation corresponds with different translations of the region Q_k which tessellate the plane, as shown in Figure 3. Looking at this Figure it is easy to see that the $2k+1$ nodes located at the north boundary are connected to the $2k+1$ nodes at the south by means of wrap-around links which are skewed k positions. The same applies to east and west boundaries.

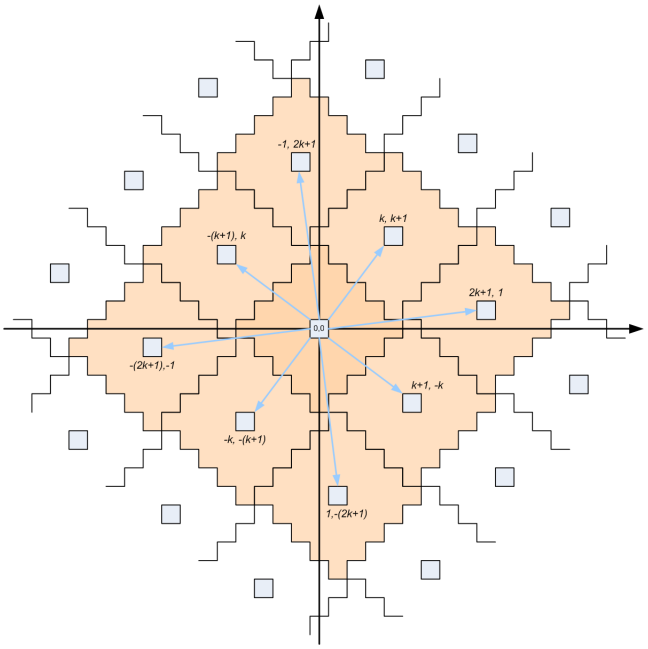


Figure 3: Plane tessellation.

4 Unicast Routing

There are many applications over dense Gaussian networks that can benefit from the two-dimensional labeling of nodes presented in the previous Section. We consider first the

problem of unicast minimal packet routing.

To send a packet from node (x, y) to node (x', y'), we need to obtain $(\Delta X, \Delta Y) = (x' - x, y' - y)$ MOD $(k, k+1)$, with $|\Delta X| + |\Delta Y|$ minimum. Once $(\Delta X, \Delta Y)$ is obtained, ΔX represents the number of links that the packet must traverse along the axis of the first coordinate and ΔY the number of links along the second coordinate's axis. Then, the network interface will produce a packet header containing two fields, ΔX and ΔY, which indicate the links to be taken in each axis; their signs indicate directions E/W and N/S directions. Routers will process the header information in the same way as in a Torus, decrementing the corresponding field header before sending the packet to the selected neighbor. A packet with $\Delta X = 0$ and $\Delta Y = 0$, will have reached its destination and will be delivered.

In order to reduce the hardware complexity of the routing record computation, we have developed a new algorithm to compute $(\Delta X, \Delta Y)$ by using only sums and comparisons. Such an algorithm will be based on the following Proposition.

Proposition 2 *Let k be a positive integer. Let $(x, y), (x', y') \in \mathbb{Z} \times \mathbb{Z}$ be such that $|x|+|y| \leq k$ and $|x'|+|y'| \leq k$. If $(\Delta X, \Delta Y) \equiv (x'-x, y'-y)$ MOD $(k, k+1)$ is such that $|\Delta X|+|\Delta Y|$ minimum, then $(\Delta X, \Delta Y) = (x' - x, y' - y) + (s_1, s_2)$ with $(s_1, s_2) \in \{(0,0), (k, k+1), (-k, -k-1), (-k-1, k), (k+1, -k), (-1, 2k+1), (1, -2k-1), (2k+1, 1), (-2k-1, -1)\}$.*

Although a detailed proof can be found in [13], the idea behind this Proposition is that the minimal path between two nodes, (x, y) and (x', y'), always results in one of nine path alternatives considering the destination node image in the nine tessellations, as illustrated in Figure 3. Given $k > 0$, we consider $(\Delta X, \Delta Y) = (x' - x, y' - y)$. Then, $(\Delta X, \Delta Y)$ is either inside the region 0 or in any of the other eight neighbor regions labeled in Figure 3 from 1 to 8. Hence, we could compute the weight of 9 integer couples and choose the one with minimum weight. Algorithm 1 describes this simple mechanism.
Just as an example of how this mechanism performs, consider again G_3 in Figure 2.

Data: (x, y): Source node; (x', y'): Destination node; k: graph diameter
Result: $(\Delta X, \Delta Y)$: routing_record$((x, y), (x', y'), k)$
begin

$\Delta X := x' - x$; $\Delta Y := y' - y$;

DO IN PARALLEL:

begin

$$\begin{cases} \Delta x_0 := \Delta x - 0; \\ \Delta y_0 := \Delta y - 0; \end{cases} \quad \begin{cases} \Delta x_1 := \Delta x - k; \\ \Delta y_1 := \Delta y - (k+1); \end{cases}$$

$$\begin{cases} \Delta x_2 := \Delta x + 1; \\ \Delta y_2 := \Delta y - (2k+1); \end{cases} \quad \begin{cases} \Delta x_3 := \Delta x + (k+1); \\ \Delta y_3 := \Delta y - k; \end{cases}$$

$$\begin{cases} \Delta x_4 := \Delta x + (2k+1); \\ \Delta y_4 := \Delta y + 1; \end{cases} \quad \begin{cases} \Delta x_5 := \Delta x + k; \\ \Delta y_5 := \Delta y + (k+1); \end{cases}$$

$$\begin{cases} \Delta x_6 := \Delta x - 1; \\ \Delta y_6 := \Delta y + (2k+1); \end{cases} \quad \begin{cases} \Delta x_7 := \Delta x - (k+1); \\ \Delta y_7 := \Delta y + k; \end{cases}$$

$$\begin{cases} \Delta x_8 := \Delta x - (2k+1); \\ \Delta y_8 := \Delta y - 1; \end{cases}$$

end

$(\Delta X, \Delta Y) := (\Delta x_i, \Delta y_i)$ such that $|\Delta x_i| + |\Delta y_i|$ is minimum;

end

Algorithm 1: Unicast Routing Algorithm in dense Gaussian networks.

Now, consider $(x, y) = (-2, -1)$ and $(x', y') = (1, 1)$. We have to compute nine possible candidates for the minimum path. As $(x' - x, y' - y) = (3, 2)$, we obtain candidates $(x' - x, y' - y) + (s_1, s_2)$, where

$$(s_1, s_2) \in \{(0,0), (3,4), (-3,-4), (-4,3), (4,-3), (-1,7), (-7,-1), (1,-7), (7,1)\},$$

Therefore, we have to choose the pair with minimum weight in the set:

$$\{(3,2), (6,6), (0,-2), (1,5), (7,-1), (2,9), (-4,1), (4,-5), (10,3)\}.$$

It is clear that $routing\_record((x, y), (x', y'), 3) = (0, -2)$, which gives us a minimal path of length 2 for reaching the node $(1, 1)$ from node $(-2, -1)$.

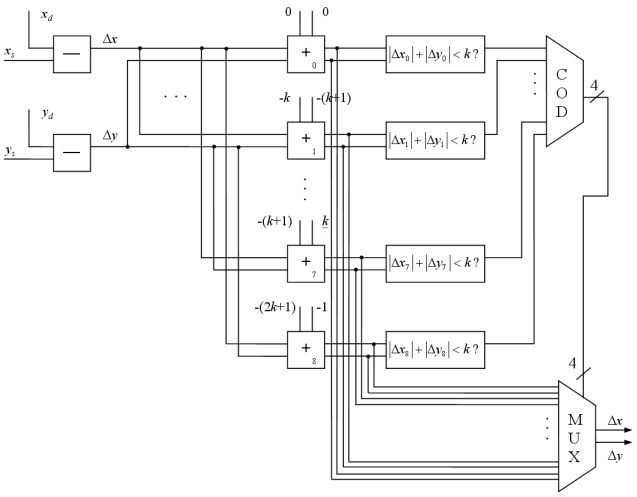


Figure 4: Routing Record Generator.

The resulting routing record generator can be easily implemented in hardware. A parallel implementation using nine adders and nine comparators will provide the fastest solution. Figure 4 presents an sketch of such a routing record generator circuit. Cheaper alternatives can also be implemented. Anyway, this routing reduces the complexity of previous mechanisms by avoiding integer divisions and it also provides an scalable implementation.

5 Broadcast Routing

In this Section we present an optimal broadcast routing for dense Gaussian networks. Efficient implementation of collective communications for parallel computing is a research topic that has received increasing attention in recent years. Broadcast communications are employed in many parallel applications such as matrix multiplication, LU factorization, Householder transformations and other basic linear algebra algorithms. Moreover, important architectural issues such as maintaining cache coherency and supporting barrier synchronization in multiprocessors may depend on the ability of the network to perform broadcasting communication [11], [14].

We will refer to Figure 5 to describe our one-to-all broadcast algorithm. In this Figure, we can identify a unitary central square in which node $(0,0)$ is located and four "discrete" right-angled triangles with identical legs of size k. We denote this special triangle as a *k-triangle*. The number of nodes of a k-triangle is $k(k+1)/2$. This geometric analysis allows us to present a dense Gaussian network of diameter k as a central node plus four k-triangular quadrants. Therefore, node $(0,0)$ has four different neighbors: $(1,0)$, $(-1,0)$, $(0,1)$ and $(0,-1)$. Each of these nodes is located on the right angle of a different k-triangular quadrant: node $(1,0)$ belongs to quadrant SE, node $(0,1)$ belongs to NE, node $(-1,0)$ belongs to NW and node $(0,-1)$ belongs to SW. From each one of these nodes located at distance 1 from node $(0,0)$ we can visit d different nodes at distance d, with $1 < d \leq k$.

We assume a router model with full-duplex links and all-port capability. Routers will support both unicast and broadcast routing, with the first header bit in every packet (B/U) indicating the class of routing service. In the case of broadcast routing, the second field in the packet header, denoted as *distance*, will be set to the network diameter, k, when the broadcast communication starts. Before each new hop, every router will decrement this field and when *distance* reaches zero, the broadcast will have finished. The third and last

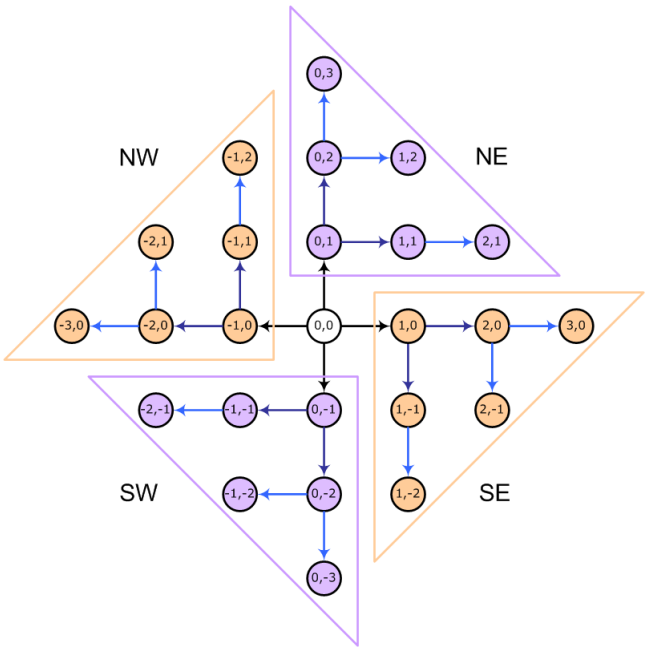


Figure 5: One-to-all Broadcast Pattern.

field in the packet header, denoted as $NSEW$, has four bits to indicate to the router the output ports to which the packet will be forwarded. We use bitmasks to deal with these bits that we denote as B_mask. The resulting header is quite compact: $\log_2 k$ plus 5 bits, nearly the same bits as needed for recording routing records for unicast traffic.

Any node starting a broadcast injects a packet to its local router with $B/U = 1$, $distance = k$ and $NSEW = 1111$. In the first step, the source node broadcasts in four directions, reaching the right angle of each k-triangle. Each of the output ports of the source node has its own bitmask, and updates the packet header according to it. For example, North output has a bitmask $B\_mask = 1010$, as it sends the packet into the NE k-triangle. The row (or column) reached by node $(0, 0)$ will continue to broadcast in both dimensions, while the other nodes will only propagate the packet along their column (or row) and updating their B_mask. For example, nodes $(0, 1)$ and $(0, 2)$ on the NE triangle broadcast to the North and East and node $(1, 1)$ only to the East, so that node $(1, 2)$ does not receive a duplicate.

Consequently, the broadcast occurs in k steps as Figure 5 reflects. In each step d, the $4d$ nodes at distance d from the source are reached with no contention. Note that the utilization of the network links is balanced, as in each step d, there are d packets traveling in each of the network quadrants. This means that it is possible to make a balanced use of the N, S, E and W network links when all nodes broadcast at once.

By using broadcast bitmasks at each output port we can obtain a simple hardware implementation. The ports B_mask is fixed, and the packet bitmask is updated on each output port it transverses. In this implementation, any received broadcast packet is consumed and sent to the outputs whose bits are set in the header field $NSEW$, and then each output port will update that header field by doing a logical AND operation with its own B_mask. Algorithm 2 describes this mechanism.

As there are no duplicates, this algorithm uses $N - 1 = 2k^2 + 2k$ links, which is the optimal number for a one-to-all broadcast operation. Besides, this algorithm is universal for every node and ends in time proportional to the diameter k, which is of order of $\sqrt{\frac{N}{2}}$. A similar broadcast in a Torus needs $\sqrt{N}$ steps. In addition, a hardware implementation of our one-to-all broadcast is much simpler than equivalent optimal mechanisms for Torus networks, such as the one proposed in [21].

6 Implementation issues

In this work, we introduce Dense Gaussian Networks as suitable topologies for modern high-end multiprocessors whose nodes are, as well, on-chip multiprocessors. We are going to consider in this Section two important architectural issues. The first one deals with the implementation of the on-chip network and the second with the inter-node network.

Data: $(B/U, distance, NSEW)$: Header of the incoming packet

$N\_out\_mask = 1010$; $S\_out\_mask = 0101$;

$E\_out\_mask = 0110$; $W\_out\_mask = 1001$

begin

if $(distance < k)$ **then**

$CONSUME$ packet;

end

if $(distance > 0)$ **then**

$distance := distance - 1$;

if $NSEW\&\&1000$ **then**

send packet to N with $(NSEW := NSEW\&\&N\_out\_mask)$;

end

if $NSEW\&\&0100$ **then**

send packet to S with $(NSEW := NSEW\&\&S\_out\_mask))$;

end

if $NSEW\&\&0010$ **then**

send packet to E with $(NSEW := NSEW\&\&E\_out\_mask))$;

end

if $NSEW\&\&0001$ **then**

send packet to W with $(NSEW := NSEW\&\&W\_out\_mask))$;

end

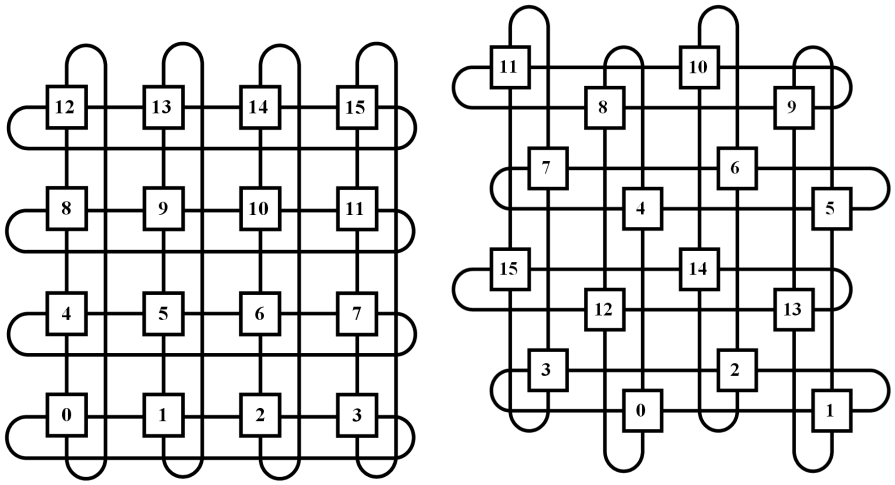
end

end

Algorithm 2: One-to-All routing algorithm in a Dense Gaussian Network.

6.1 Folded Dense Gaussian Network

Dense Gaussian Networks, as 2D-Tori, are mesh-like topologies with wrap-around links, whose lengths grow with the network size. While internal links are supposed to have unitary length, wrap-around links in square Torus grow as $\sqrt{N}$, where N is the number of nodes. As a consequence, an on-chip implementation can be negatively affected by this unbalance. In the case of the Torus, the Folded Torus presented in Figure 6 is a solution to equalize the network links by increasing the wire length to 2. With the same aim, a new layout for Dense Gaussian Networks was proposed in [18], obtaining as a result a maximum wire length bounded by $\sqrt{5}$.



(a) 4×4 Torus (b) 4×4 Folded Torus

Figure 6: 4×4 Torus and Folded Torus

The algorithm begins with an *initial layout* of the network, arranged as follows. Nodes are located in $2k + 1$ rows and $k + 1$ columns. An example for the case of G_4 with $N = 41$ is shown in Figure 7. The first row contains a single node (node $(0, 0)$, at the end of row 1). Each node will have four links, two of them joining with the node above and the one on the left of this. Vertical links will increase the coordinate in Y, while diagonal links will

increase the coordinate in X, with all the operations MOD $(k, k+1)$ as defined in Section 3. This procedure allows us to label all the nodes. Remaining links, shown in grey in the Figure, can be obtained using the adjacency pattern of DGN.

The algorithm that maps a DGN into a bounded link layout is presented in Algorithm 6.1. Given a row with nodes $\{1, 2, \ldots n\}$ two shuffle transformations which map every node location onto a different one on the same row are defined in the following way:

- Shuffle A:

$$x' = 2x - 1 \qquad if\ x \leq (n+1)/2.$$
$$x' = 2n - 2x + 2 \quad if\ x > (n+1)/2.$$

- Shuffle B :

$$x' = 2x \qquad if\ x < (n+1)/2.$$
$$x' = 2n - 2x + 1 \quad if\ x \geq (n+1)/2.$$

After using different rotations and shuffles of the rows and columns of the network, we obtain a mapping of the dense Gaussian networks with no links larger than $\sqrt{5}$. As an example, this algorithm converts the initial layout into the final layout in Figure 7.

An important property of any network-on-chip layout is the number of different metal layers required to arrange all its links. This parameter will have a significant impact on its cost. In the case of DGNs, four planes are enough to lay all the network links without cutting links, as shown in [18]. Even more, in most of the area, except the upper and lower links, three planes are enough.

6.2 Hierarchical Gaussian Networks

High Performance Computing systems are already being designed on the idea of multi-CMPs, this is, systems built by joining together several Chip-Multiprocessors (CMPs), such as [1]. While Gaussian Networks appear as a competitive option for the intra-chip

Data: t: Diameter of the network to map
Step 1 or Initial layout: Arrange the $N = 2k^2 + 2k + 1$ nodes in $2k + 1$ rows $(1, ..., 2k + 1)$ in an *initial layout* as defined above.
Step 2 or Row rotation and shuffle:
-For rows $1 \leq i \leq k + 1$, apply a rotation $\left\lfloor \frac{i-1}{2} \right\rfloor$ and then apply an A shuffle to odd rows and a B shuffle to even ones;
-For rows $k + 2 \leq i \leq 2k + 1$, apply a rotation $\left\lfloor \frac{i}{2} \right\rfloor$ and then apply a B shuffle to odd rows and an A shuffle to even ones;
Step 3 or Column shuffle A: Shuffle all columns according to shuffle A.

Algorithm 3: Mapping Algorithm

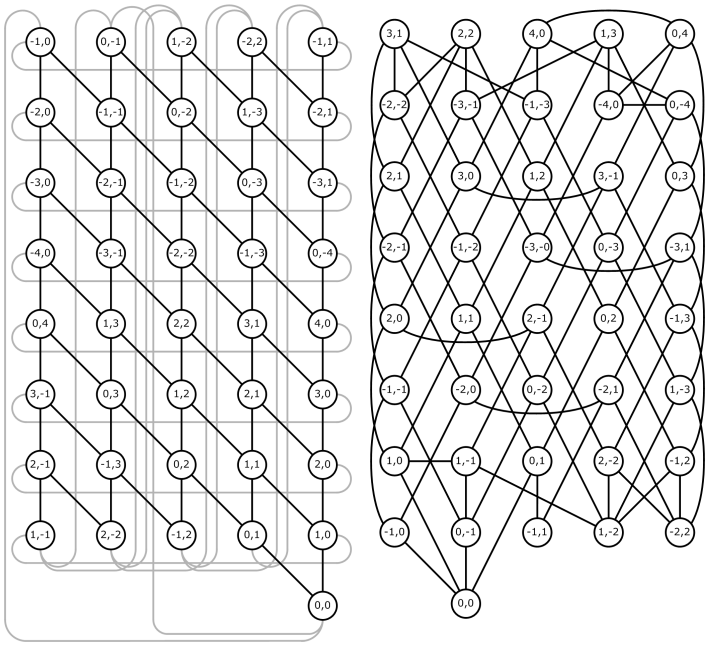
interconnect, a hierarchical approach is needed to interconnect different CMPs. Hence, it is necessary exploring new networks whose topological properties match the new requirements imposed by these emerging architectures. We explore in this Subsection hierarchical Gaussian networks as possible candidates for implementing such two-level interconnection networks.

Next, we define the two-level hierarchical Gaussian network, while it can be generalized to any number of levels.

Definition 3 *Given k a positive integer we define the* ***Two-Level Hierarchical Gaussian Network*** $\mathbf{HG}_k$ *of G_k as follows:*

- $Q_k \times Q_k := \{((x, y), (x', y')) \mid (x, y), (x', y') \in Q_k\}$ *is the set of nodes and*
- *A node $((x, y), (x', y'))$ is adjacent to a node $((x_0, y_0), (x'_0, y'_0))$ if and only if $(x, y) = (x_0, y_0)$ and $d((x', y'), (x'_0, y'_0)) = 1$ or $(x', y') = (x'_0, y'_0)$ and $d((x, y), (x_0, y_0)) = 1$, where d is the distance in G_k.*

An intuitive visualization of how to build this network is to take N dense Gaussian networks of N nodes and join their centers following the adjacency pattern of a dense Gaussian network of N nodes. A simple example with $k = 3$ and $N = 25$ can be seen in Figure 8.



(a) Initial layout for $N = 41$ (b) Final layout for $N = 41$

Figure 7: Folded Dense Gaussian Network.

Some of the wrap-around links are omitted for the sake of simplicity. Thus, we have that $\mathbf{HG}_k$ has N^2 nodes and $2N^2 + 2N$ links, where $N = k^2 + (k+1)^2$. Also, it is clear that the diameter of this structure is $3k$. This is neither a regular graph, as we have nodes of degree four and eight, nor a vertex-symmetric graph. We denote the links in lower level of hierarchy as **base** links, while the links in the higher level are denoted as **express** links. Unicast routing in this hierarchical network can be obtained from a direct generalization of the Proposition 2 of Section 4, while broadcasting should also consider the different levels of the network. This hierarchical Networks can be easily applied to the design of multi-CMP systems, being the lower level the on-chip Network, and the higher level the inter-chip network.

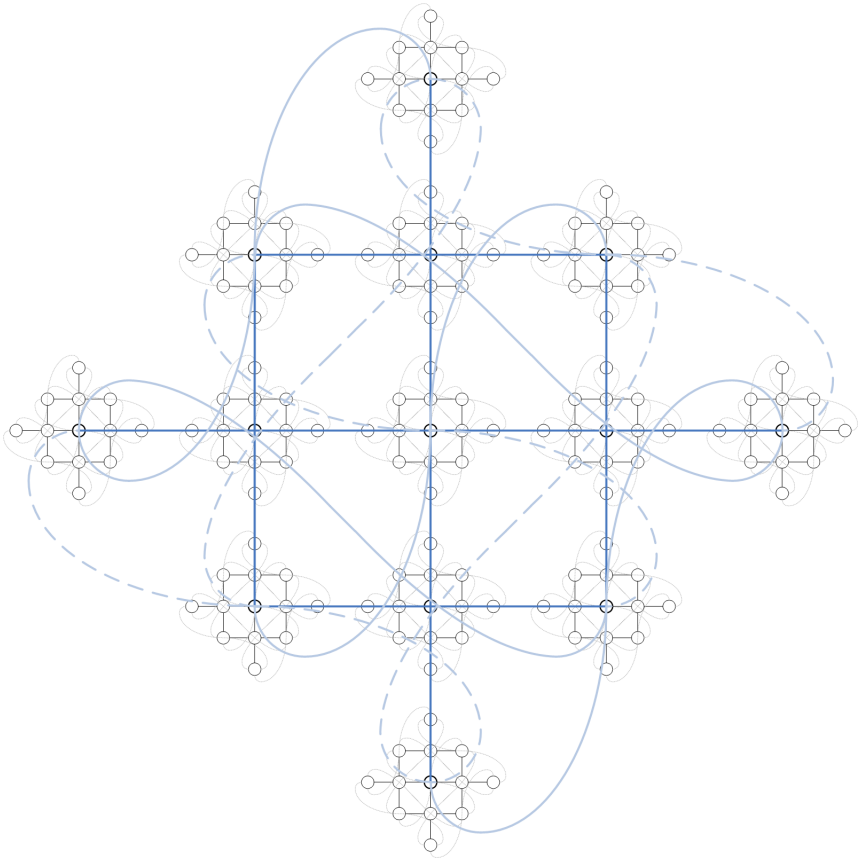


Figure 8: Hierarchical Gaussian Network $\mathbf{HG}_3$

7 Conclusions

This paper introduces Dense Gaussian Networks as a suitable regular two-dimensional topology for on-chip networks. This mesh-like topology reaches the maximum number of nodes for a given diameter, meaning that it improves diameter and average distance against any other two-dimensional mesh-based topology. This paper translate these topological advantages into real network gains by presenting and analyzing different architectural issues that makes Dense Gaussian Networks attractive for on-chip parallel computing.

First of all, a new two-dimensional node's labeling of the networks explored in this work has been proposed. In this way, the two-dimensional nature of these networks can be ex-

ploited which facilitates their analysis. Based on this new labeling we have proposed both optimal unicast and broadcast routing schemes that make an efficient use of the network resources. In addition, a smart layout for a two-dimensional VLSI network implementation which equalizes the length of all the network links has been also introduced. Such new layout makes this topology suitable for embedded on-chip systems. Finally, a hierarchical design presenting an extended network to connect multiple on-chip systems has been also described.

In conclusion, the overall properties of Dense Gaussian Networks outdo other well-known topologies such as Tori, by just rearranging some of the network links. Thus, these networks appear as a clear alternative to be considered for the design of future parallel systems.

References

[1] L. A. Barroso et al. "Piranha: A Scalable Architecture Based on Single-Chip Multiprocessing". 27^{th} Annual International Symposium on Computer Architecture, pp. 282-293, June 2000.

[2] R. Beivide, E. Herrada, J. L. Balcázar and A. Arruabarrena. "Optimal Distance Networks of Low Degree for Parallel Computers". IEEE Transactions on Computers, Vol. C-40, No. 10, pp. 1109-1124, 1991.

[3] R. Beivide, E. Herrada, J. L. Balcazar and J. Labarta. "Optimized Mesh-Connected Networks for SIMD and MIMD Architectures". 14^{th} Annual International Symposium on Computer Architecture, pp. 163-169, 1987.

[4] J.-C. Bermond, G. Illiades and C. Peyrat. "An Optimization Problem in Distributed Loop Computer Networks". 3rd Interantional Conference on Combinatorials Mathematics. New York Academy of Sciences, pp. 1-13, 1985.

[5] M. Blumrich et al. "Design and Analysis of the BlueGene/L Torus Interconnection Network". IBM Research Report, RC23025 (W0312-022), December 3, Computer Science, 2003.

[6] F. T. Boesch and J. Wang. "Reliable Circulant Networks with Minimum Transmission Delay". IEEE Transactions on Circuit and Systems. Vol. 32, pp. 1286-1291, 1985.

[7] Z. Cvetanovic. "Performance Analysis of the Alpha 21364-based HP GS1280 Multiprocessor". 30^{th} Annual International Symposium on Computer Architecture. pp. 218-228, 2003.

[8] M. A. Fiol, J. L. Yebra, I. Alegre and M. Valero. "A Discrete Optimization Problem in Local Networks and Data Alignment". IEEE Transactions on Computers, Vol. 36, No. 6, pp. 702-713, 1987.

[9] M. R. Mullins, J. D. Bingham, M. D. Hill, A. J. Hu, M. M. Martin and D. A. Wood. "Improving Multiple-CMP Systems Using Token Coherence". 11th International Sysmposium on High Performance Computer Architecture, pp. 328-339. 2005.

[10] R. Mullins, A. West and S. Moore. "Low-Latency Virtual-Channel Routers for On-Chip Networks". 31th International Sysmposium on Computer Architecture, pp. 188-197. 2004.

[11] M. M. K. Martin, M. D. Hill and D. A. Wood. "Token Coherence: Decoupling Performance and Correctness". 30^{th} Annual International Symposium on Computer Architecture, pp. 182-193, 2003.

[12] C. Martínez, R. Beivide, J. Gutierrez and E. Gabidulin. "On the Perfect t-Dominating Set Problem in Circulant Graphs and Codes over Gaussian Integers". Proceedings of the 2005 IEEE International Symposium on Information Theory (ISIT'05). Adelaide, Australia. Septiembre, 2005.

[13] C. Martínez, E. Vallejo, M. Moretó, R. Beivide y M. Valero, Hierarchical Topologies for Large-scale Two-level Networks, XVI Jornadas de Paralelismo. Granada, Spain, September 2005.

[14] J. Mellor-Crummey and M. Scott. "Algorithms for Scalable Synchronization on Shared- Memory Multiprocessors". ACM Transactions on Computer Systems 9(1), pp. 2165, February 1991.

[15] V. Puente, C. Izu, J.A. Gregorio, R. Beivide, J.M. Prellezo and F. Vallejo. "Rearranging Links to Improve the Performance of Parallel Computers: The Case of Midimew Networks". International Conference on Supercomputing, ICS'2000, pp. 44-53, 2000.

[16] V. Puente, J.A. Gregorio, F. Vallejo and R. Beivide. "Immunet: A Cheap and Robust Fault-Tolerant Packet Routing Mechanism". 31^{th} Annual International Symposium on Computer Architecture, pp. 198-209, 2004.

[17] J.M. Tendler et al. "Power4 System Microarchitecture". IBM Journal of Research and Development, 46(1), 2002.

[18] E. Vallejo, R. Beivide and C. Martínez. "Practicable Layouts for Optimal Circulant Graphs". Euromicro Conference on Parallel, Distributed and Network-based Processing, Switzerland, February 2005.

[19] C. K. Wong and D. Coppersmith. "A Combinatorial Problem Related to Multimodule Memory Organizations". Journal of the ACM, Vol. 21, No. 3, pp. 392-402, 1974.

[20] Y. Yang, A. Funashi, A. Jouraku, H. Nishi, H. Amano and T. Sueyoshi. "Recursive Diagonal Torus: An Interconnection Network for Massively Parallel Computers". IEEE Transactions on Parallel and Distributed Systems, Vol. 12, No. 7, July 2001.

[21] Y. Yang and J. Wang. "Efficient All-to-All Broadcast in All-Port Mesh and Torus Networks". 5^{th} International Symposium on High Performance Computer Architecture, Florida, 1999.