Parallel Construction of Edge-Disjoint Node-Independent Spanning Trees in Dense Gaussian Networks

Z. Hussain[1], **B. AlBdaiwi**[1], **and H. AboElfotoh**[1]
[1]Computer Science Department, Kuwait University, Kuwait

Abstract— *This paper presents parallel constructions for edge-disjoint node-independent spanning trees in dense Gaussian networks using Tree-based and Propagation-based methods. These spanning trees are useful in secure message distribution and fault-tolerant broadcasting. The presented constructions are simulated with and without the presence of faulty nodes. The results show that due to the tree cut in the Tree-based construction the maximum average number of steps required to construct the trees decreases when one or more faulty nodes are present. On the contrary, the presence of faulty nodes in the Propagation-based method increases the paths' lengths and hence increases the constructions' steps.*

Keywords: Parallel Algorithms, Circulant Graphs, Gaussian Networks, Spanning Trees, Independent Spanning Trees.

1. Introduction

An interconnection network consists of nodes and links where each link connects two nodes. A node has one or more processing elements, a local memory, and a communication module. The nodes in the interconnection network communicate by exchanging messages through the links. Each interconnection network has topological properties that specifies its characteristics such as the total number of nodes, the total number of links, the diameter, the average distance... etc. There are varieties of interconnection network topologies designed and used in parallel computing. Some of the most popular networks are Hypercubes [1][2], Generalized Hypercubes [3], Twisted Cubes [4], Cube Connected Cycle [5], k–ary n–cube [6], and Torus [7].

The topology of Gaussian networks, which is known to be efficient, was introduced in [8][9]. Gaussian networks are symmetric and each node in the network is of degree 4, i.e., 4-regular network. They have a topology similar to that of Torus networks in terms of symmetry, regularity, the number of nodes, and the number of links. The advantage of Gaussian networks' topology over Torus networks' topology is that their diameter is less, which makes Gaussian networks potential alternatives for Torus networks. Gaussian networks are called dense when it contains the maximum number of nodes for a given diameter. These networks are reviewed in Section 2. Further studies and advantages of Gaussian networks can be refered to in [10][11][12][13][14].

In parallel systems, usually an interconnection network is represented by a graph $G(V, E)$, where V, the set of vertices, represents the network nodes; and E, the set of edges, represents the network links. A spanning tree $T(G) = (V, E')$, simply T, is an acyclic subgraph of G that connects all the vertices V by the subset of edges E', $E' \subseteq E$. A set of spanning trees $T_1, T_2, \ldots, T_n$ rooted at vertex r are said to be node-independent if, for any vertex v, all paths connecting r and v in all spanning trees T_j, for $j = 1, 2, \ldots, n$, are pairwise internally disjoint. Two or more spanning trees are said to be edge-disjoint if they share no common edges. That is, each edge used in one spanning tree is not used in any other spanning tree. Independent spanning trees that are edge-disjoint are called Edge-Disjoint Node-Independent Spanning Trees (EDNISTs).

There are useful applications for independent spanning trees in interconnection networks. One application is secure message distribution [15][16]. Consider a network with n independet spanning trees rooted at node r. Then, we can divide a message into n packets such that each packet is sent by the root node r to its destination using a distinct spanning tree. That is, the destination node receives the n packets and all other nodes receive at most one of the n packets. Another application is node fault-tolerant broadcasting [17][18]. For example, consider a network with n independent spanning trees rooted at node r and at most $n-1$ faulty nodes. Then, r can broadcast a message to all non-faulty nodes in the network with the existence of maximum $n-1$ faulty nodes. Thus, there exist at least one spanning tree that delivers the message to all other non-faulty nodes in the network. EDNISTs can be used in secure message distribution and node fault-tolerant broadcasting as they are independent spanning trees. Furthermore, they can be used in edge and/or node fault-tolerate broadcasting because of a similar reasoning to the one mentioned above.

There are some previous studies on independent spanning trees in Gaussian networks. EDNIST constructions in dense Gaussian networks has been introduced in [19]. Whereas, the problem of finding four independent spanning trees, where the edges are not necessarily disjoint, in dense Gaussian networks has been studied in [20]. In this paper, we present parallel construction algorithms for EDNIST in dense Gaussian networks.

This paper is organized as follows. Gaussian networks and some preliminaries are reviewed in Section 2. Parallel

constructions of EDNISTs in dense Gaussian networks are described in Section 3. Simulation results are discussed in Section 4. Finally, the paper is concluded in Section 5.

2. Preliminaries

In this section, we review some definitions and properties of graphs. Then, we review the topological properties of Gaussian networks.

Let $G(V, E)$ be a graph where V is the set of vertices and E is the set of the edges. Two vertices u and v are said to be neighbors if both are directly connected via an edge, which is represented by (u, v). A path $P(u, v)$ from vertex u to vertex v in G is a finite sequence of edges which connect a sequences of distinct vertices starting from vertex u and ending at vertex v. Two paths $P_1(u, v)$ and $P_2(u, v)$ are said to be independent if their intermediate vertices are mutually disjoint. A spanning tree T of graph G is a subgraph of G that is a tree contains all the vertices of G. i.e., $T(V', E') \subseteq G(V, E)$ where $V' = V$ and $E' \subseteq E$. Two or more spanning trees T_j, for $j = 1, 2, \ldots, n$, rooted at vertex r are called independent spanning trees if the paths from the root vertex r to any other vertex u in all trees are independent. Furthermore, independent spanning trees are called EDNISTs when the edge sets are pairwise disjoint, i.e., for all trees $T_j(V, E'_j)$, for $j = 1, 2, \ldots, n$, we have $E'_p \cap E'_q = \phi$ for all $p \neq q$ such that $1 \leq p \leq n$ and $1 \leq q \leq n$. The distance between two vertices u and v in graph G is known to be the number of edges in a shortest path and is defined as $D(u, v) = min\{|P(u, v)| \ : \ u, v \in V\}$ where $|P(u, v)|$ is the length of path $P(u, v)$. That is, we choose the path with minimum length from all possible paths between u to v. The diameter k of the graph is known as the shortest distance between two most farthest vertices in graph G.

Gaussian networks are planar graphs. They can be modeled as a graph $G_\alpha(V, E)$ generated by $\alpha = a + bi$ such that $0 \leq a \leq b$, where $V = \mathbb{Z}[i]_\alpha$ is the vertex set modulo α, which represents the nodes in the network; and $E = \{(A, B) \in V \times V : (A - B) \equiv \pm 1, \pm i \ mod \ \alpha\}$ is the edge set, which represents the network links. They are based on quotient rings of Gaussian integers. The Gaussian integers [9][21][22] are the subset of complex numbers that contain integer real, x, and integer imaginary, y, parts; they are defined as $\mathbb{Z}[i] = \{x + yi : x, y \in \mathbb{Z}\}$ where $\mathbb{Z} \in \{0, 1, 2, \ldots\}$ and $i = \sqrt{-1}$. Let $\alpha = a + bi \neq 0$ be a Gaussian integer, the total number of nodes in the network is called norm of α, $N(\alpha) = a^2 + b^2$, and it is equal to the number of elements in the residue class modulo α; there are many shapes for this network which can be found in [23]. Gaussian networks are 4-regular symmetric networks. Each node in the network is labeled as $x + yi$. Two nodes are adjacent if the difference between them modulo α is equal to ± 1 or $\pm i$, i.e., the distance from A to B is 1. The distance between two nodes is defined as $D_\alpha(A, B) = min\{|x| + |y| \mid (A - B) \equiv x + yi \ (mod \ \alpha)\}$. Gaussian networks are called dense Gaussian networks if they contain the maximum number of nodes at a certain distance, usually their generator is $\alpha = a + bi$ such that $b = a + 1$. Thus, the number of nodes at distance s is 1 or $4s$, respectively, for $s = 0$ or $s > 0$. It can be concluded that the diameter of dense Gaussian networks is $k = a$.

Gaussian networks are symmetric. They contain two types of links: regular and wraparound links. The regular links are links that connect two nodes within the grid of the network. Whereas, the wraparound links connect two nodes on the boundary of the grid. Simply, a node A is connected to a node B via wraparound link if $(A \pm 1$ or $A \pm i)$ $mod \ \alpha \equiv B$.

For example, Figure 1 illustrates the Gaussian network generated by $\alpha = 4+5i$. The solid lines represent the regular links, whereas, the dotted lines represent the wraparound links. Furthermore, Figure 2 shows the nodes that are out of the grid boundary with their equivalent nodes that are within the grid of the network. For instance, the node $3i$ is connected to the node $4i$, which is equivalent to the node -3, via the $+i$ link.

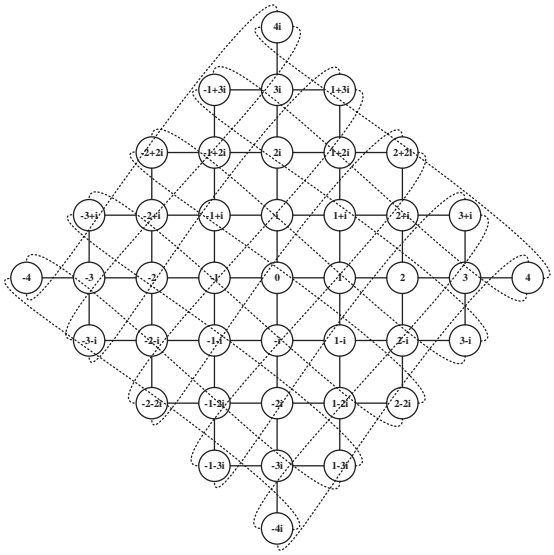

Fig. 1: Gaussian network generated by $\alpha = 4 + 5i$. Dotted lines are wraparound links.

3. The Edge-Disjoint Node-Independent Spanning Trees

This section discusses the parallel construction of edge-disjoint node-independent spanning trees in dense Gaussian networks. First, we illustrate the partitions of a Gaussian network, which are helpful in constructing the spanning trees. Second, we describe the parent and child nodes of each node in the network. Finally, we give a parallel

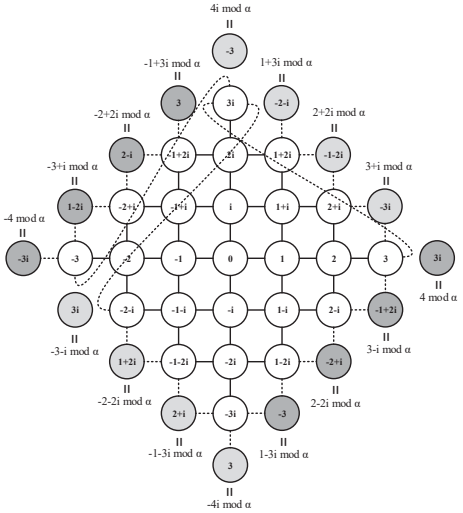

Fig. 2: Wraparound links in Gaussian network generated by $\alpha = 3 + 4i$

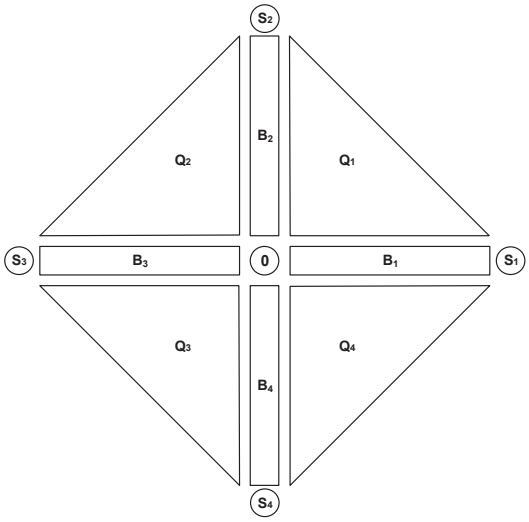

Fig. 3: Gaussian network partitions

construction algorithm for these trees using both Tree-based and Propagation-based methods.

The dense Gaussian network, generated by $\alpha = a + bi$ where $b = a + 1$, can be partitioned into 12 disjoint subset as shown in Figure 3, excluding the node 0 (i.e., the root node). These disjoint subsets are as follows:

$$Q_1 = \{x + yi \mid x > 0,\, y > 0\},$$
$$Q_2 = \{x + yi \mid x < 0,\, y > 0\},$$
$$Q_3 = \{x + yi \mid x < 0,\, y < 0\},$$
$$Q_4 = \{x + yi \mid x > 0,\, y < 0\},$$
$$B_1 = \{x + yi \mid 0 < x < k,\, y = 0\},$$
$$B_2 = \{x + yi \mid x = 0,\, 0 < y < k\},$$
$$B_3 = \{x + yi \mid -k < x < 0,\, y = 0\},$$
$$B_4 = \{x + yi \mid x = 0,\, -k < y < 0\},$$
$$S_1 = \{x + yi \mid x = k,\, y = 0\},$$
$$S_2 = \{x + yi \mid x = 0,\, y = k\},$$
$$S_3 = \{x + yi \mid x = -k,\, y = 0\},$$
$$S_4 = \{x + yi \mid x = 0,\, y = -k\}$$

Table 1 shows the trees parent and child nodes for each node $v = x + yi$ in the network. The root node has no parent node and has two child nodes in the first tree T_1, which are $+1$ and $+i$; and it has two other child nodes in the second tree T_2, which are -1 and $-i$. By using Table 1, it is possible to construct the two edge-disjoint node-independent spanning trees in dense Gaussian networks. For example, The two trees for $\alpha = 4 + 5i$ are drawn in Figures 4 and 5. The disjointness and independency of these trees were proven in [19].

Table 1: Parent and child nodes in T_1 and T_2 for a given node $v = x + yi$

		Tree 1		Tree 2	
Set	Condition	Parent	Child	Parent	Child
Q_1	$x > 0,\, y > 0$	-1	$+1$	$+i$	$-i$
Q_2	$x < 0,\, y > 0$	$+i$	$-i$	-1	$+1$
Q_3	$x < 0,\, y < 0$	-1	$+1$	$+i$	$-i$
Q_4	$x > 0,\, y < 0$	$+i$	$-i$	-1	$+1$
B_1	$0 < x < k,\, y = 0$	-1	$+1, -i$	$+i$	–
B_2	$x = 0,\, 0 < y < k$	$-i$	$+i, +1$	-1	–
B_3	$-k < x < 0,\, y = 0$	$+i$	–	$+1$	$-1, -i$
B_4	$x = 0,\, -k < y < 0$	-1	–	$+i$	$-i, +1$
S_1	$x = k,\, y = 0$	-1	$-i$	$+i$	–
S_2	$x = 0,\, y = k$	$-i$	$+1$	$+i$	–
S_3	$x = -k,\, y = 0$	$+i$	–	$+1$	$-i$
S_4	$x = 0,\, y = -k$	-1	–	$+i$	$-i$

3.1 Tree-Based Construction Method

The Tree-based construction method works as follows. The root node, i.e. node 0, sends a message to its children in Tree i. The message indicates the index of the tree being constructed. Accordingly, each receiving node identifies its parent and children as per Table 1, and forwards the received message to its children. Algorithms 1 and 2 describe the Tree-based construction in dense Gaussian networks generated by $\alpha = a + bi$.

Node 0 in Algorithms 1 and 2 is considered as the root node. Any node in the network can be a root node by mapping it to node 0 and mapping all other nodes accordingly. Thus, the Tree-based construction algorithms is not restricted to node 0 being always the root node.

3.2 Propagation-Based Construction Method

The Propagation-based method construction is based on the broadcasting concept. The root node sends its address,

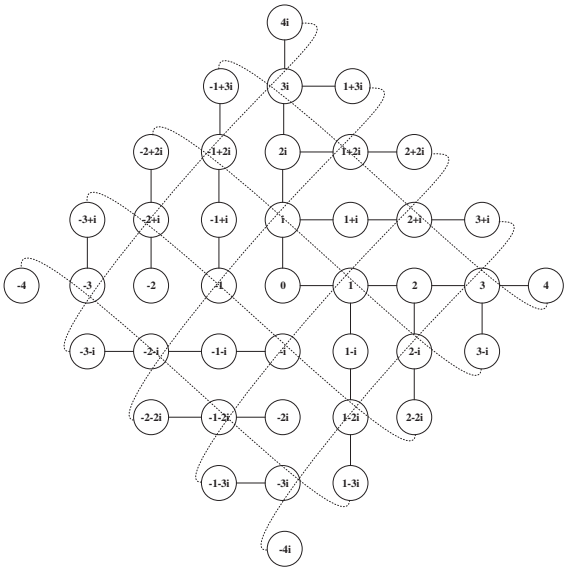

Fig. 4: The tree $T_1, k = 4$

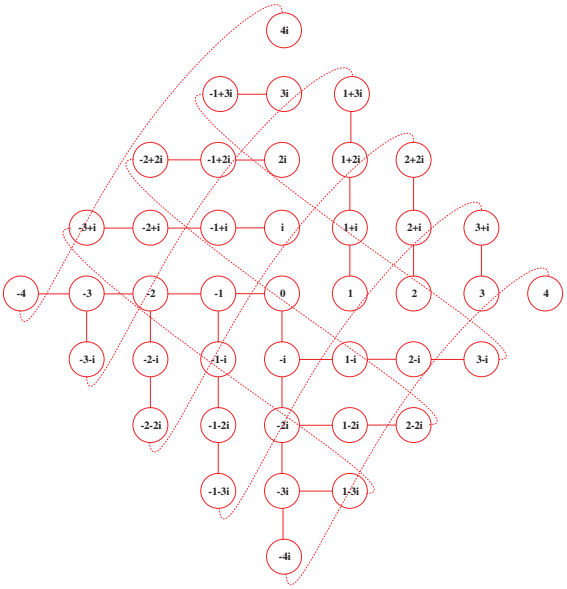

Fig. 5: The tree $T_2, k = 4$

addr, to all its neighboring nodes. Based on Table 1, each receiving node identifies its parents and children in both trees, and forwards the message to all its neighboring nodes, except the node it already received the message from.

Algorithms 3 and 4 describe the construction in details.

4. Simulation

We have simulated both constructions using NetworkX simulator. We have generated nine Gaussian networks with

Algorithm 1 InitialNode: Tree-based construction initial step

1: $root \leftarrow 0$
2: $root$.Tree[1].Child[1] $\leftarrow root + 1$
3: $root$.Tree[1].Child[2] $\leftarrow root + i$
4: $root$.Tree[2].Child[1] $\leftarrow root - 1$
5: $root$.Tree[2].Child[2] $\leftarrow root - i$
6: Send through $+1$ packet ($root$, 1)
7: Send through $+i$ packet ($root$, 1)
8: Send through -1 packet ($root$, 2)
9: Send through $-i$ packet ($root$, 2)

Algorithm 2 IntermediateNode: Current node processing based on the received packet ($parent$, j)

1: Let C be the current working node of form $x + yi$
2: $c = 1$ be the child index.
3: C.Tree[j].Parent $\leftarrow$ *parent*
4: C.Tree[j].Child[1] $\leftarrow$ NIL
5: C.Tree[j].Child[2] $\leftarrow$ NIL
6: **for each** childNode of C in Table 1 for tree j **do**
7: C.Tree[j].Child[c] $\leftarrow C +$ childNode
8: C sends through childNode packet (C, j)
9: $c \leftarrow c + 1$
10: **end for**

Algorithm 3 RootNode(S): Trees construction from node S

1: addr $\leftarrow$ address of S.
2: S sends through $+1$ packet (addr)
3: S sends through $+i$ packet (addr)
4: S sends through -1 packet (addr)
5: S sends through $-i$ packet (addr)

Algorithm 4 Node process based on the received (addr)

1: **if** (addr) has been received before **then**
2: Ignore the message and exit.
3: **end if**
4: Compute the relative address of the current node based on the received (addr).
5: Match the current node relative address with the entries of Table 2 to determine the current node parent and children links in all trees: T_1 and T_2.
6: Send (addr) to all neighbor nodes except the one it was already received from.

different sizes. For each network size, we ran four distinct types of simulations. The first type simulates both constructions in fault-free networks. Whereas, the second, third, and fourth types of simulations considers networks with 1, 2, and 3, respectively. Tables 2 and 3 show the average maximum number of steps required to construct the edge-disjoint node-independent spanning trees for the networks using Tree-based and Propagation-based algorithms, respectively. We also measured the average number of steps required for each construction.

The data presented in Tables 2 and 3 are illustrated, in respective order, in Figures 6 and 7.

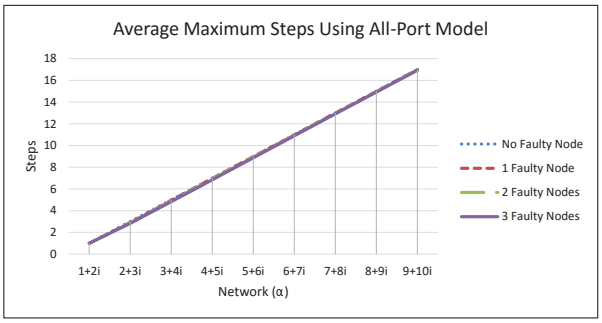

Fig. 6: Average maximum steps based on tree-based method

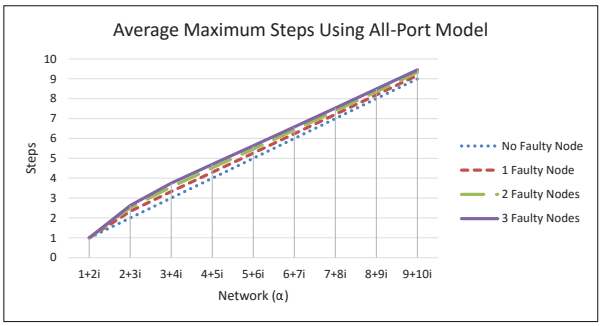

Fig. 7: Average maximum steps based on propagation-based method

Obviously, the Propagation-based method outperforms the Tree-based method in terms of the average number and average maximum number of steps required for the constructions. This is simply due to the higher degree of parallelism of the Propagation-based method over the Tree-based method.

5. Conclusions

In this paper, we have briefly discussed some background and preliminaries about Gaussian networks. Then, we have partitioned the dense Gaussian network into 12 subsets. Based on these subsets we designed two construction methods for edge-disjoint node-independent spanning trees, namely Tree-based and Propagation-based methods. Both constructions were simulated using NetworkX simulator on different network sizes and different faulty nodes settings. The simulation results show the superiority of the Propagation-based method over the Tree-based method in terms of the constructions' maximum and average numbers of steps; this is simply due to the higher degree of parallelism in the Propagation-based method when compared to the Tree-based method.

References

[1] J. P. Hayes and T. Mudge, "Hypercube supercomputers," *Proceedings of the IEEE*, vol. 77, no. 12, pp. 1829–1841, 1989.

[2] J. P. Hayes, T. N. Mudge, Q. F. Stout, S. Colley, and J. Palmer, "Architecture of a hypercube supercomputer." in *ICPP*, 1986, pp. 653–660.

[3] L. N. Bhuyan and D. P. Agrawal, "Generalized hypercube and hyperbus structures for a computer network," *IEEE Transactions on computers*, vol. 100, no. 4, pp. 323–333, 1984.

[4] P. A. Hilbers, M. R. Koopman, and J. L. Van de Snepscheut, "The twisted cube," in *International Conference on Parallel Architectures and Languages Europe*. Springer, 1987, pp. 152–159.

[5] F. P. Preparata and J. Vuillemin, "The cube-connected cycles: a versatile network for parallel computation," *Communications of the ACM*, vol. 24, no. 5, pp. 300–309, 1981.

[6] B. Bose, B. Broeg, Y. Kwon, and Y. Ashir, "Lee distance and topological properties of k-ary n-cubes," *IEEE Transactions on Computers*, vol. 44, no. 8, pp. 1021–1030, 1995.

[7] W. J. Dally and C. L. Seitz, "The torus routing chip," *Distributed computing*, vol. 1, no. 4, pp. 187–196, 1986.

[8] C. Martinez, E. Vallejo, R. Beivide, C. Izu, and M. Moreto, "Dense Gaussian networks: Suitable topologies for on-chip multiprocessors," *International Journal of Parallel Programming*, vol. 34, pp. 193–211, 2006.

[9] C. Martínez, R. Beivide, E. Stafford, M. Moretó, and E. M. Gabidulin, "Modeling toroidal networks with the gaussian integers," *IEEE Transactions on Computers*, vol. 57, no. 8, pp. 1046–1056, 2008.

[10] B. Bose, A. Shamaei, and M. Flahive, "Higher dimensional gaussian networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 9, pp. 2628–2638, Sept 2016.

[11] M. Flahive and B. Bose, "The topology of Gaussian and Eisenstein-Jacobi interconnection networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 8, pp. 1132–1142, August 2010.

[12] Z. Hussain, "Better traffic distribution one-to-all broadcast in higher dimensional gaussian networks," *The Journal of Supercomputing*, vol. 71, no. 12, pp. 4381–4399, 2015.

[13] A. Touzene, "On all-to-all broadcast in dense gaussian network on-chip," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 1085–1095, April 2015.

[14] Z. Zhang, Z. Guo, and Y. Yang, "Efficient all-to-all broadcast in gaussian on-chip networks," *IEEE Transactions on Computers*, vol. 62, no. 10, pp. 1959–1971, 2013.

[15] A. A. Rescigno, "Vertex-disjoint spanning trees of the star network with applications to fault-tolerance and security," *Information Sciences*, vol. 137, no. 1, pp. 259–276, 2001.

[16] J.-S. Yang, H.-C. Chan, and J.-M. Chang, "Broadcasting secure messages via optimal independent spanning trees in folded hypercubes," *Discrete Applied Mathematics*, vol. 159, no. 12, pp. 1254–1263, 2011.

[17] A. Itai and M. Rodeh, "The multi-tree approach to reliability in distributed networks," *Information and Computation*, vol. 79, no. 1, pp. 43–59, 1988.

[18] M. Krishnamoorthy and B. Krishnamurthy, "Fault diameter of interconnection networks," *Computers & Mathematics with Applications*, vol. 13, no. 5, pp. 577–582, 1987.

[19] B. AlBdaiwi, Z. Hussain, A. Cerny, and R. Aldred, "Edge-disjoint node-independent spanning trees in dense gaussian networks," *The Journal of Supercomputing*, pp. 1–19, 2016. [Online]. Available: http://dx.doi.org/10.1007/s11227-016-1768-x

Table 2: Average maximum steps based on tree-based method and using all-port model.

Network Size	1+2i	2+3i	3+4i	4+5i	5+6i	6+7i	7+8i	8+9i	9+10i
No Faulty Node	1	3	5	7	9	11	13	15	17
1 Faulty Node	1	2.916	4.958	6.975	8.983	10.988	12.991	14.993	16.994
2 Faulty Nodes	1	2.878	4.894	6.921	8.939	10.952	12.96	14.967	16.972
3 Faulty Nodes	1	2.836	4.824	6.852	8.877	10.898	12.914	14.927	16.937

Table 3: Average maximum steps based on propagation-based method and using all-port model.

Network Size	1+2i	2+3i	3+4i	4+5i	5+6i	6+7i	7+8i	8+9i	9+10i
No Faulty Node	1	2	3	4	5	6	7	8	9
1 Faulty Node	1	2.333	3.333	4.3	5.266	6.238	7.214	8.194	9.177
2 Faulty Nodes	1	2.515	3.565	4.52	5.47	6.426	7.388	8.355	9.327
3 Faulty Nodes	1	2.627	3.751	4.698	5.631	6.577	7.53	8.489	9.453

[20] Z. Hussain, B. AlBdaiwi, and A. Cerny, "Node-independent spanning trees in gaussian networks," in *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*. Las Vegas, NV, USA: The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2016, pp. 24–29.

[21] G. H. Hardy and E. M. Wright, *An introduction to the theory of numbers*. Oxford University Press, 1979.

[22] K. Huber, "Codes over gaussian integers," *IEEE Transactions on Information Theory*, vol. 40, no. 1, pp. 207–216, 1994.

[23] C. J. P. J. H. Jordan, "Complete residue systems in the gaussian integers," *Mathematics Magazine*, vol. 38, no. 1, pp. 1–12, 1965. [Online]. Available: http://www.jstor.org/stable/2688007