Perfect Difference Network-based Parallel Computation using a Geometry Driven Approach

Anurag Tiwari
AKS University
Satna, MP, India 485001

Pinki Sharma
AKS University
Satna, MP, India 485001

Akhilesh A. Waoo
AKS University
Satna, MP, India 485001

ABSTRACT
Parallel and distributed computing systems are greatly affected by the structure of the underlying network topology and geometry. In this paper, a new method of parallel computing inspired by geometry is suggested. The Perfect Difference Network (PDN), which is based on perfect difference sets, is employed to develop an efficient parallel matrix multiplication algorithm. To this end, the geometry of the underlying architectures is considered to investigate the connection and communication between the processors. In this regard, it is shown that geometry plays an important role in parallel computing. The suggested algorithm is coded using MPI in a distributed multiprocessor framework. The performance analysis of the algorithm is carried out through experiments performed on networks with varying sizes (N = 7, 13, and 26). Performance indicators like execution time, communication delay, average hop count, and load balancing efficiency are used to evaluate the algorithm's performance. It is shown from the simulation studies that although PDN performs well in terms of routing efficiency and balance, performance is more influenced by communication delay as the number of processors increases.

General Terms
Parallel Computing, Distributed Computing, Interconnection Networks, Network Topology, Geometric Computing Models, Performance Evaluation, High-Performance Computing.

Keywords
Communication Overhead, Distributed Systems, Geometric Properties, Interconnection Networks, Load Balancing, MPI, Parallel Algorithm, Parallel Computing, Perfect Difference Network (PDN), Scalability.

1. INTRODUCTION
With the ever-growing exponential rate of improvement in high-performance computation and data-intensive algorithms, there has been a necessity for the development of efficient parallel and distributed systems. In such scenarios, the performance of the system not only depends upon its computational efficiency but also upon the effectiveness of the communication process among the processors. The network structure of the network that is being used in such cases becomes extremely important for defining factors such as latency, scalability, and reliability [3], [5]. Network structures such as Mesh, Torus, Hypercube, and Tree topology have been used extensively because of their symmetry and scalability properties [4], [6]. Nevertheless, each of these network structures comes with its own drawbacks depending on the scale at which it is implemented.

The geometrical attributes of computer network architectures, such as connectivity, location, and communication channel length, play an important role in the performance of the architecture. Spatial relationships in processor architectures have attracted increasing interest since they make it possible to analyze communication behavior in distributed systems [13]. Models based on graph theory can be used to model such architectures, making it possible to analyze connectivity and other related aspects of the architecture [14]. Past studies have shown that there is potential for improving the efficiency of communication channels by designing architecture topology [3], [15].

Among the new interconnection networks that have gained considerable attention is the Perfect Difference Network (PDN), which possesses a distinctive architecture built around the Perfect Difference Set (PDS) concept, which is rooted in combinatorics [1], [2]. The PDNs have a highly symmetric and structured topology with minimal node degrees, yet they possess short paths for communication. This enables efficient scalability and routing in comparison to the traditional networks [1]. Moreover, the PDNs have proven to perform efficiently in terms of communication attributes [10], [11].

Nevertheless, in spite of these strengths, most of the literature concerning PDNs mainly concentrates on modeling, structural analysis, or simulations to evaluate their effectiveness. Not many efforts have been undertaken in incorporating the geometric layout of PDNs in the design of parallel algorithms and assessing their effectiveness through real-world applications. Although a few studies have conducted investigations on PDNs based on theoretical models like the PRAM model [12], there is still no application in popular platforms like MPI [16].

In order to solve the problem stated above, this paper introduces a novel parallel algorithmic strategy that relies on geometry, specifically, the use of the Perfect Difference Network. An MPI (mpi4py) parallel implementation of the PDN-based algorithm utilizing modular arithmetic routing via perfect difference sets will be presented. Performance of the introduced methodology is experimentally estimated using several configurations and analyzed in terms of various performance metrics, including the execution time, communication delay, number of hops required, and efficiency of load balancing. The main contributions of this paper include introducing a geometric parallel algorithm based on the PDN approach and the utilization of perfect difference set-based communications within a distributed computing system.

2. RELATED WORK
Network Topology is one of the important factors in terms of influence on the performance of parallel and distributed computations. Traditional topologies include Mesh, Torus, Hypercube, and Tree, and there have been numerous research studies dedicated to such kinds of geometries. The work done

by Leighton [4], Siegel [5], and Dally and Towles [3] emphasizes the significance of network topology for improving communication and system performance. Other works show that the design of the network is crucial to reducing latencies and increasing the throughput of the computation. Therefore, we see that the network geometry plays an important role in improving performance.

Introduction: The Perfect Difference Network (PDN) has proven to be an effective approach in comparison with conventional approaches because of its special design that involves the use of Perfect Difference Sets (PDS) [2]. PDN is defined as an interconnection topology with high symmetry, low diameter, and regularity of node degrees. These properties have been analyzed through comparative research [10], as well as in the studies conducted by Sharma et al. [11], [12]. Further investigation of PDN properties is carried out through the application of the graph theory [14] and the principle of spatial computing [13].

Despite the advances described above, the current literature lacks experimental implementations. Specifically, there are no studies where a PDN-based geometric routing is implemented in any parallel algorithm and experimentally evaluated using distributed computation models like MPI [16]. Clearly, there is a need for developing geometry-driven parallel algorithms where the concept of PDN is applied, along with an implementation and experimental evaluation of the latter. It is precisely this gap that is addressed through the design of the PDN-based parallel algorithm proposed in the current paper.

3. METHODOLOGY

3.1. Research Framework

The suggested research adopts a geometric paradigm for parallel processing, incorporating the attributes of interconnection structures in the development of algorithms. There are four key phases in this methodology: (i) modeling of interconnection systems using graph-theoretic techniques, (ii) geometric considerations related to communication issues, (iii) formulation of a parallel algorithm based on PDNs, and (iv) experimentation via MPI. The use of graph-theoretic representation allows formal analysis of the attributes of processors and their interactions [14]. The entire framework is guided by previous research stressing the significance of network topology in communication issues [3], [6].

3.2. Modeling of Perfect Difference Network (PDN)

A PDN network is constructed using Perfect Difference Sets (PDS) that come from finite projective geometry [2]. A PDN with N nodes can be made when the following holds:

$$N = q^2 + q + 1$$

Here, q is either prime or a prime power. The difference set D consists of q+1 elements, all of which ensure that all the non-zero differences modulo N are distinct [1]. Nodes in PDN are connected to other nodes within their neighborhood, and the neighborhood is determined by the following:

$$N(i) = \{(i \pm d_k) \bmod N\}, d_k \in D$$

This modular connectivity ensures uniform node degree, symmetric network structure, and low network diameter (typically ≤ 2). These properties make PDN highly suitable for efficient communication in parallel systems [1], [10].

Following Fig. 1 shows an example of the PDN structure for N = 7, which has been constructed on the basis of the perfect difference set D = {1,2,4}. Each node is connected to six other nodes through modular arithmetic, and therefore, the whole network is fully symmetric and completely connected, ensuring efficient communication channels.

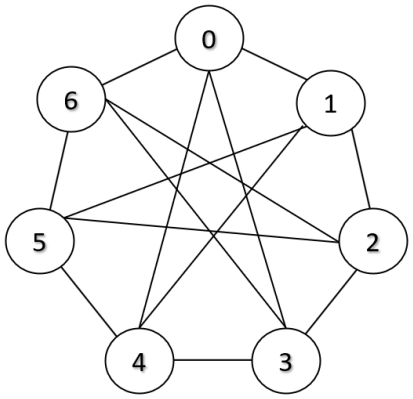

Fig 1: Perfect Difference Network Topology for N = 7

3.3. Geometric Properties and Communication Model

Performance of the parallel system is significantly dependent upon the geometry of networks, which includes aspects such as node degrees, diameter, and connectivity. Geometric properties of the PDN allow for effective communication with minimal hops. Communication between nodes operates through the principle of modular arithmetic routing: $dest = (i + d_k) \bmod N \; and \; source = (i - d_k) \bmod N$. These routing techniques guarantee effective communication using short routes. Earlier researchers have highlighted the significance of topology-based communication in improving communication efficiency in distributed systems [3], [6].

3.4. Geometry-Driven Parallel Algorithm Design

A parallel matrix multiplication algorithm has been devised using PDN connectivity. Input matrices are split among N processors, such that each processor has. P_i undertakes local computations and exchanges information with its neighbors using the routing strategy prescribed by PDN. At each iteration k, the parameters used for routing are chosen from the difference set,

$$d_k = D[k \bmod \; | \, D \, |]$$

The computation is performed as follows.

$$C_i = C_i + A_i \times B_i$$

and then communications as:

$$A_i \rightarrow (i + d_k) \bmod N, B_i \rightarrow (i + d_{k+1}) \bmod N$$

This geometry-driven approach of communications ensures effective information exchange with low communication costs.

Below mentioned process of the PDN-based parallel algorithm implementation is depicted in the following flow chart provided in Fig. 2. Every processor initializes local data and calculates with the help of communication between the processor and its neighboring nodes via modular routing in terms of a perfect difference set**.**

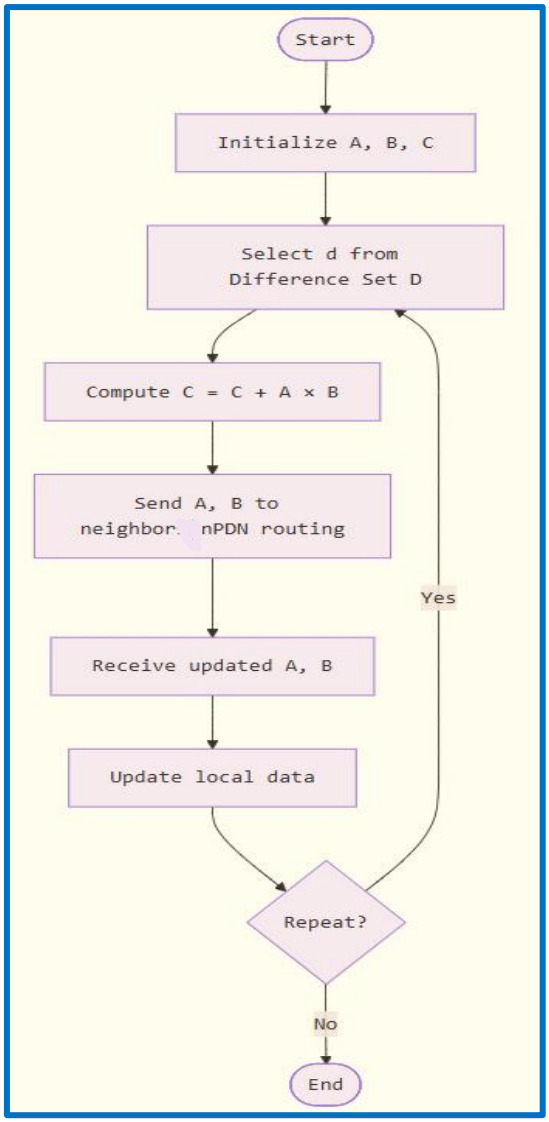

Fig 2: Flow-chart of Geometry-Driven Parallel Algorithm Design

3.5. Implementation Using MPI

The algorithm proposed here is coded in the form of a software solution that employs MPI (Message Passing Interface), which is a commonly used open interface standard for distributed parallel computing [16]. This software has been created with the use of the Python programming language together with the mpi4py library, in which an MPI process is equal to a PDN node and the process number equals the node number i. The communication between PDN nodes takes place through non-blocking sends (Isend) and receives (Recv), while the synchronization among MPI processes takes place by means of the Waitall function.

3.6. Performance Metrics

In order to determine the performance of the above algorithm, some metrics need to be calculated. The first metric is execution time, which is the maximum processing time at each node. This metric determines how much time the entire parallel system takes to complete its tasks. The second metric is communication time, and it is the total time taken by the system for the sending and receiving of data. The third metric is average hop count, which is the mean hop count in the system, and it provides information about the efficiency of the system's communication. Finally, the load distribution efficiency metric calculates the ratio of the minimum to maximum execution times, reflecting the efficiency of load distribution in the system. Table 1 below summarizes all the performance metrics used in this paper.

Table 1. Performance Metrics for PDN

Metric	Formula	Description
Execution Time	$T_{exec} = \max(T_i)$	Maximum computation time among all nodes
Communication Time	$T_{comm} = \sum(t_{send} + t_{recv})$	Total time spent in sending and receiving messages
Average Hop Count	$H_{avg} = \frac{\text{Total hops}}{N}$	Average number of communication hops per node
Load Distribution Efficiency	$E = \frac{T_{\min}}{T_{\max}}$	Ratio indicating how evenly the load is distributed across nodes

3.7. Experimental Setup

Tests of the algorithm are done for a range of PDN topologies to study the performance characteristics of the algorithm in different structures. This involves testing the algorithm using N=7 and D={1,2,4}, and also N=13 and D={1,3,9}. Apart from these baseline cases, more advanced topologies are also tested to determine the performance characteristics of the algorithms under different larger network structures. The experiments that are described in Table 2 below involve determining the performance of the algorithm under the impact of the geometrical network structure.

Table 2: PDN Configurations Used for Evaluation

Configuration	Number of Nodes (N)	Generator Set (D)	Purpose
Config 1	7	{1, 2, 4}	Baseline evaluation of small-scale PDN
Config 2	13	{1, 3, 9}	Medium-scale performance analysis
Extended Configs	Variable	Variable	Scalability and large-scale behavior analysis

4. RESULT AND DISCUSSION

4.1. Experimental Results

The performance of the proposed PDN algorithm is analyzed for various numbers of network nodes, and the results are shown in Table 3. From the table, it can be seen that with increasing values of N from N=7 to N=26, there is a significant increment in both the execution and communication time, demonstrating the presence of more overheads in computing and communication in large-scale systems. An increasing trend is also observed in terms of hop count with an increasing network size, depicting an increase in the number of

communication pathways in the PDN network. On the other hand, load efficiency varies in the case of different networks, where N=7 exhibits more efficiency than N=13 and even N=26.

Table 3: Performance Metrics Obtained Result

N	Execution Time (sec)	Communication Time (sec)	Avg Hop Count	Load Efficiency
7	0.003745	0.003220	14.00	0.7378
13	0.035244	0.035016	26.00	0.4255
26	0.373552	0.373060	52.00	0.6222

4.2. Analysis of Results

4.2.1. Execution Time Behavior

Execution time grows proportionately with the number of processors as shown in Fig 3 below. Although normally, the parallel execution decreases the time required for computations, the execution time grows because a major share is taken by the communication costs in the distributed system. Execution time goes up from 0.0037 sec (N=7) to 0.3735 sec (N=26), which means that with increasing numbers of processors and thus larger systems, the cost of communication and synchronization rapidly grows [3], [6].

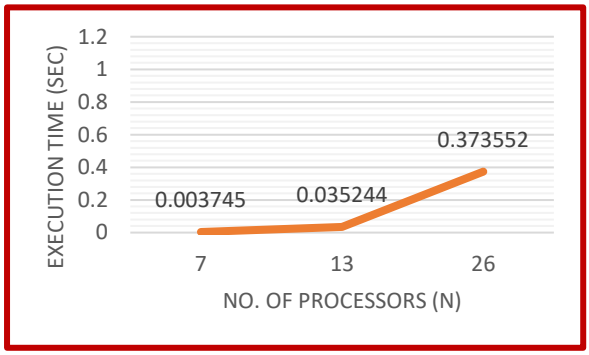

Fig 3: Execution Time Behavior of PDN

4.2.2. Communication Overhead

It can be observed from the output of the experiment that the time taken to communicate is very close to that of the execution time in all cases as shown in Fig 4 below. In the case of large network sizes (N=13 and N=26), communication becomes almost equal to the execution time. It shows that the algorithm under discussion is bound by communication rather than computation. PDN ensures good route efficiency due to its low diameter.

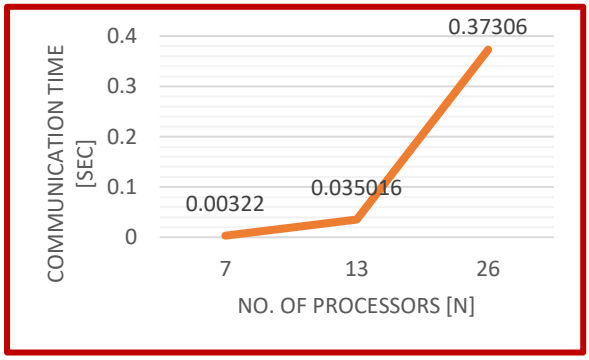

Fig 4: Communication Overhead of PDN

4.2.3. Hop Count Analysis

The average hop count rises linearly with the size of the network: 14→26→52 as shown in Fig 5 below. This is because of the growing number of communication stages as the system grows larger. But because of PDN's geometric properties, each communication stage usually entails direct modular routing, which makes the length of the path shorter than in conventional architectures [1], [10].

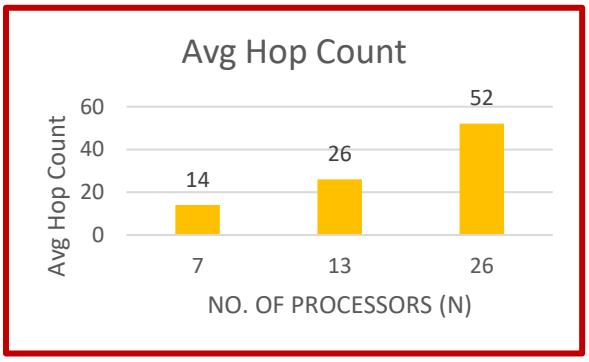

Fig 5: Hop Count Analysis of PDN

4.2.4. Load Distribution Efficiency

Load distribution efficiency varies across configurations:

- N=7: High efficiency (0.7378)
- N=13: Significant drop (0.4255)
- N=26: Moderate improvement (0.6222)

This deviation demonstrates that although PDN provides uniform connectivity, runtime load balancing is influenced by communication delay and process coordination. The decline at N = 13 shows unbalanced load distribution caused by communication delay, while the partial recovery at N = 26 signifies better utilization of network topology as shown in Fig 6 below.

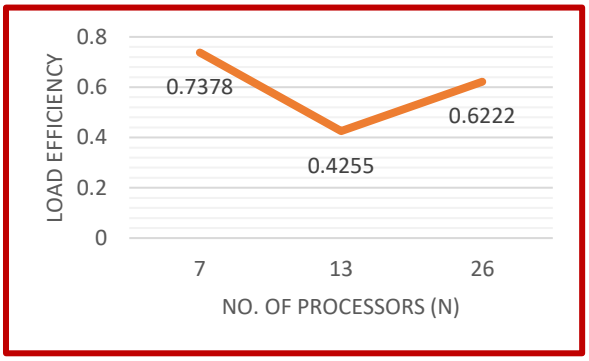

Fig 6: Load distribution efficiency of PDN

4.2.5. Scalability Discussion

From the analysis, it is evident that although the PDN algorithm is very scalable, scalability is achieved at the expense of low performance because of the message overhead involved. This implies that the larger the number of processes employed, the greater the cost incurred during message passing. Nevertheless, PDN is scalable and routable.

4.2.6. Overall Discussion

The experimental results support the benefits that accrue from using geometric features in designing parallel algorithms, particularly in relation to routing and connectivity. It is evident, however, that communication cost is equally important for the distributed computing environment. The PDN algorithm offers an advantage in reducing the hop count and providing even connectivity; nevertheless, the increase in communication cost with system size becomes significant.

Based on the experiment results, it can be observed that the execution time is mainly governed by the communication overhead, which escalates rapidly with an increasing number of processors. While the hop count shows a linear relationship with respect to the system size, it is very efficient owing to the well-defined routing scheme used in the Perfect Difference Network (PDN). The load balancing on processors is fair; nevertheless, it is subject to communication overhead and synchronization delay. In summary, it can be concluded that the PDN is much more efficient structurally than conventional interconnection networks.

5. CONCLUSION

In this paper, the authors introduce a parallel computing scheme driven by geometry using the Perfect Difference Network (PDN). First, a theoretical examination was carried out on the geometrical properties of the interconnection networks and identified that factors like connectivity, degree of nodes, and routing play a vital role in determining the effectiveness of parallel and distributed computing environments. Based on the findings above, a new parallel scheme was developed, making use of routing using perfect difference sets through modular routing. This parallel algorithm was executed using MPI (mpi4py) in a multi-process distributed environment. Different schemes were evaluated for different values (N = 7, 13, and 26), and the performance was measured using execution time, latency, hops, and other related factors.

From the results, it is clear that the PDN structure is an efficient and symmetric network connection structure. This implies a reduced number of communication channels and an even balancing of workloads. On the other hand, the experiment also indicates that the overall performance of the system is constrained by communication. Communication overhead becomes very high when more processors are involved. Despite this problem, it is evident that the PDN structure can scale well with respect to communication overheads and routing overheads. It means that the structure is appropriate for designing algorithms that are geometry-based. In general, the research proves that geometry-based algorithm design can lead to increased efficiency of communications.

6. FUTURE WORK

While the efficiency demonstrated in this research is high, there are many aspects that require additional research in order to increase the efficiency of parallel computations in PDN networks. Some examples include improving communication methods to reduce the overhead costs of communications using advanced methods like message aggregation and asynchronous scheduling. Other methods include the use of hybrid network topologies that will allow the integration of various network topologies to increase efficiency. One such example might be the combination of the PDN topology with hypercubes or toruses. Scalability is another crucial aspect, and this can be obtained by scaling the current design to become an HPC cluster implementation for large data sets. In addition, the geometric approach used in this research can also be applied in other parallel algorithms, such as sorting algorithms, graph-based algorithms, and machine learning algorithms. The fault tolerance and reliability of the proposed designs are also very essential aspects that need more research, especially regarding the performance of PDN when there are faulty nodes or links in the network. Moreover, integrating PDN-based designs in modern technologies like cloud computing [8], edge computing [7],blue Gene supercomputer [9] and IoT-based distributed computing could give interesting results.

7. ACKNOWLEDGMENTS

The authors would like to express their sincere gratitude to AKS University, Satna, Madhya Pradesh, India, for providing the academic environment and research facilities necessary for conducting this work. The authors are thankful to the faculty members and researchers of the Department of Computer Science for their valuable suggestions and technical discussions throughout the course of this study.

8. REFERENCES

[1] B. Parhami and M. Rakov, “Perfect difference networks and related interconnection structures for parallel and distributed systems,” IEEE Transactions on Parallel and Distributed Systems, vol. 16, no. 8, pp. 714–724, Aug. 2005.

[2] J. Singer, “A theorem in finite projective geometry and some applications to number theory,” Transactions of the American Mathematical Society, vol. 43, no. 3, pp. 377–385, 1938.

[3] W. J. Dally and B. Towles, Principles and Practices of Interconnection Networks. San Francisco, CA, USA: Morgan Kaufmann, 2004.

[4] F. T. Leighton, Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes. San Mateo, CA, USA: Morgan Kaufmann, 1992.

[5] H. J. Siegel, Interconnection Networks for Large-Scale Parallel Processing: Theory and Case Studies. Lexington, MA, USA: Lexington Books, 1990.

[6] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, “Performance evaluation and design trade-offs for network-on-chip interconnect architectures,” IEEE Transactions on Computers, vol. 54, no. 8, pp. 1025–1040, Aug. 2005.

[7] I. Foster and C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure. San Francisco, CA, USA: Morgan Kaufmann, 2003.

[8] M. Armbrust et al., “A view of cloud computing,” Communications of the ACM, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[9] S. Kumar, A. Mamidala, D. Faraj, et al., “Introduction to Blue Gene/L supercomputer architecture,” IBM Journal of Research and Development, vol. 49, no. 2/3, pp. 195–212, 2005.

[10] R. K. Katare and N. S. Chaudhari, “A comparative study of hypercube and perfect difference network for parallel and distributed systems,” International Journal of Computer Applications, vol. 1, no. 21, pp. 1–5, 2010.

[11] P. Sharma, R. Begum, R. Katare, and A. Waoo, “Exploring the interplay between communication complexity and network density in PDNs,” ShodhKosh: Journal of Visual and Performing Arts, vol. 5, no. 5, 2024.

[12] P. Sharma, S. Tripathi, R. Katare, and A. Waoo, "PRAM-based algorithm for perfect difference network analysis," ShodhKosh: Journal of Visual and Performing Arts, vol. 5, no. 1, 2024.

[13] S. Shekhar, S. K. Feiner, and W. G. Aref, "Spatial computing: Issues, trends, and challenges," Communications of the ACM, vol. 63, no. 6, pp. 72–81, 2020.

[14] J. L. Gross and J. Yellen, Graph Theory and Its Applications. Boca Raton, FL, USA: CRC Press, 2005.

[15] L. Benini and G. De Micheli, "Networks on chips: A new SoC paradigm," IEEE Computer, vol. 35, no. 1, pp. 70–78, Jan. 2002.

[16] Träff, Jesper Larsson. "Distributed Memory Parallel Systems and MPI." *Lectures on Parallel Computing*. Cham: Springer Nature Switzerland, 2026. 171-287.