TMP UNIVERSAL JOURNAL OF RESEARCH AND REVIEW ARCHIVES

VOLUME 4 | ISSUE 2 | YEAR 2025 | APRIL – JUNE 2025

RECEIVED DATE	ACCEPTED DATE	PUBLISHED DATE
07/03/2025	29/03/2025	15/04/2025

Article Type: Research Article

Available online: www.tmp.twistingmemoirs.com

ISSN 2583-7214

A NOVEL COMMUNICATION-INFRASTRUCTURE-BASED DYNAMIC ROUTING ALGORITHM FOR NETWORK-ON-CHIP TO IMPROVE LOCAL AND GLOBAL CONGESTION AWARENESS

[1*]Mohammad Khalaji, [1]Amin Mehran Zadeh

[1]Department of Engineering Computer, Dezfoul Branch, Islamic Azad University, Dezfoul, Iran.

Corresponding Author: Mohammad Khalaji

ABSTRACT

Network-on-Chip (NoC) architectures integrate all essential components of a computing system onto a single chip, enabling efficient communication among various processing units. These architectures typically encompass processors, memory modules, and input/output interfaces, facilitating seamless interaction among heterogeneous components. NoCs can incorporate a diverse range of functional units, including central processing units (CPUs), graphics processing units (GPUs), Bluetooth, Wi-Fi, and other peripheral modules, thereby enhancing system scalability and performance. The efficiency of data transmission in NoCs heavily depends on routing algorithms, which play a crucial role in managing communication latency, congestion, and overall network reliability. Routing algorithms play a fundamental role in managing data flow across various components within a Network-on-Chip (NoC) architecture, ensuring efficient and reliable communication. These algorithms are responsible for determining optimal data transmission paths while addressing key challenges such as congestion, latency, and resource utilization. Given the high density of interconnections in NoC-based systems, an effective routing strategy is essential to minimize communication interference, reduce power consumption, and enhance overall data transfer efficiency. By optimizing these factors, routing algorithms contribute significantly to improving system performance, increasing energy efficiency, and ensuring seamless on-chip communication. These algorithms play a pivotal role in enhancing the overall system performance and optimizing the efficiency of various on-chip components. In Network-on-Chip (NoC) architectures, incorporating global congestion awareness into the routing decision-making process is crucial for selecting the most efficient path to the destination. In this study, a novel communication subsystem is introduced to facilitate the acquisition of global congestion information, alongside the development of an intelligent routing algorithm that utilizes both local and global congestion data to optimize path selection. The proposed approach aims to improve system performance by enhancing throughput and reducing latency. Simulation and evaluation results obtained using the Noxim simulator demonstrate the superior efficiency of

the proposed method compared to existing routing algorithms, highlighting its effectiveness in optimizing data transmission and network reliability.

Keywords: Network-on-Chip (NoC), Routing, Communication Infrastructure, Local Congestion, Global Congestion

INTRODUCTION

With the continuous advancement of Network-on-Chip (NoC) architectures, efficient data routing has emerged as a critical challenge in their design, where traffic congestion and latency significantly impact overall performance. In such networks, data exchanged between various processing modules may encounter transmission bottlenecks due to network congestion and propagation delays, potentially degrading communication efficiency and system reliability. Addressing these challenges requires the development of intelligent routing strategies capable of optimizing data flow while minimizing congestion and delay. To address these challenges, developing robust and efficient traffic- and latency-aware routing algorithms is essential for optimizing data transmission in Network-on-Chip (NoC) architectures. These algorithms utilize real-time network congestion and delay information to intelligently manage data flow, effectively mitigating issues such as transmission interference, minimizing latency, and improving overall network performance. By dynamically adjusting to varying network conditions, such approaches enhance communication efficiency, ensuring seamless and reliable data transmission across NoC systems.

Congestion management represents a significant challenge in Network-on-Chip (NoC) systems, as the increasing number of components and the volume of data transmitted within the chip elevate the likelihood of interference and congestion (Zhou et al., 2023). Effective solutions for managing congestion in NoC networks often involve the use of intelligent routing algorithms, which are capable of dynamically selecting optimal paths for data transmission based on the current state of the network. These algorithms are designed to minimize delays and reduce interference, thereby improving overall network performance. Additionally, the utilization of alternative or dynamically adjusted paths for concurrent data transfers helps alleviate congestion and enhances the overall efficiency of the communication process. This strategy aids in selecting alternative paths in response to congestion. Efficient bandwidth management within Network-on-Chip (NoC) systems can be accomplished by allocating dedicated bandwidth to each path or by dynamically increasing bandwidth for paths experiencing higher traffic. Furthermore, the implementation of intelligent arbitration algorithms, which can dynamically allocate resources to components and make real-time decisions during congestion, is crucial for effective resource utilization. The use of predictive models and algorithms to estimate the likelihood of congestion allows for proactive measures, such as rerouting or optimal resource allocation, to be applied in advance. Traffic management solutions, including traffic control, packet prioritization, and chip load balancing, are essential for load distribution and minimizing interference. Moreover, integrating various approaches and employing a combination of algorithms and techniques can further enhance congestion management within NoC systems, improving overall network performance and efficiency

In their research on bandwidth-based multipath routing for custom Network-on-Chip (NoC) architectures, Lee et al. (2022) emphasized that the exponential growth of large-scale data and artificial intelligence has compelled computational platform architectures to evolve toward heterogeneous multicore designs to enhance energy efficiency. A customized NoC, capable of supporting a variety of connections, is crucial for addressing the traffic demands of modern heterogeneous multicore System-on-Chip (SoC) architectures, where asymmetric data access is essential for optimal performance. In a separate study, Joshi and Thakur (2023) reported that Network-on-Chip (NoC) architectures have been identified as a reliable solution to address the

challenges of flexibility and scalability in multicore architectures, issues that were prevalent in earlier generations of System-on-Chip (SoC) designs. One of the key components in this context is the routing of data packets through the network of IP cores. Advanced routing mechanisms typically involve a trade-off between two crucial performance metrics: throughput and latency. Optimizing one often leads to the performance of the other becoming a limiting factor. This research endeavors to propose a comprehensive and optimized routing algorithm aimed at overcoming this fundamental challenge.

In the study conducted by Gogoi et al. (2023), it was highlighted that intensive inter-core communication in multi-core architectures imposes an excessive workload on system components, which often leads to performance degradation and increased error rates. The Network-on-Chip (NoC) inherits these fault occurrences from multi-core architectures, affecting both the links and routers within the communication infrastructure. This research specifically examines faults occurring in NoC routers and introduces a fault-aware routing approach aimed at mitigating their impact. The proposed routing strategy dynamically determines the optimal routing path based on the health status of neighboring routers, thereby enhancing system reliability and performance. This research introduces a proactive fault-aware routing approach. In the study by Narayana et al. (2023), it was emphasized that congestion in Network-on-Chip (NoC) architectures occurs under heavy traffic loads, resulting in inefficient bandwidth utilization and severely impairing system performance. To mitigate this issue, the study proposes a lightweight machine learning-based technique designed to predict congestion by extracting traffic-related features at each destination and labeling them through a novel time-reversed approach. The labeled data is then leveraged to develop a low-overhead, interpretable decision tree model, which is subsequently employed for real-time congestion control, enhancing overall network efficiency and system reliability.

Manzoor and Mir (2022) reported that the Network-on-Chip (NoC) has emerged as a highly effective solution to meet the ever-increasing communication demands of large-scale System-on-Chip (SoC) architectures. The overall performance and efficiency of NoCs are heavily influenced by the routing mechanism used for data transmission. However, designing an efficient routing algorithm presents several challenges, among which deadlock remains one of the most significant. Deadlock is a pervasive and complex issue in NoC systems, and implementing deadlock-free routing algorithms essential for ensuring seamless and reliable network operation. Additionally, the study conducted by Fang et al. (2022) highlighted that three-dimensional Network-on-Chip (3D NoC) architectures, which rely on Through-Silicon Via (TSV) technology for interconnecting multiple chip layers, exhibit thermally heterogeneous conductivity across different silicon layers. This thermal disparity poses considerable challenges for thermal management, potentially degrading system performance and reliability. To alleviate the severity of thermal issues, data packet distribution is typically guided by either traffic load information or historical temperature data. However, relying solely on traffic or temperature data is insufficient to effectively resolve thermal challenges. To address this limitation, this study introduces a Scalable Traffic- and Thermal-Aware Routing (STTAR) approach, which integrates both real-time traffic load and thermal conditions into the routing process. The STTAR mechanism operates through a twofold strategy. First, it dynamically adjusts the input and output buffer lengths of each router based on traffic load conditions, thereby limiting routing resources in overheated regions and effectively regulating temperature rise. Second, it employs a sophisticated scoring mechanism that considers both temperature levels and the availability of free buffer slots, preventing data packets from being directed toward high-temperature or congested areas. This approach enhances the efficiency and rationality of selecting optimal routing paths. By incorporating both traffic load distribution and thermal management into the routing process, STTAR provides an adaptive and efficient solution for improving the performance and reliability of Network-on-Chip (NoC) architectures.

In the study conducted by Vazifedunn Doon et al. (2023), the authors highlighted that owing to the advantages offered by Network-on-Chip (NoC) architectures, they are rapidly advancing and poised to replace other forms of System-on-Chip (SoC) designs. While various factors contribute to the enhancement of NoC performance, numerous challenges must be addressed during the design process, one of the most significant being network congestion and its subsequent effects on system performance and efficiency. To tackle these challenges, several routing algorithms have been proposed, which account for the impact of congestion and aim to manage its complexities. However, considering the substantial influence of overhead costs on performance and overall system efficiency, it is essential to carefully address these factors when designing advanced NoC systems. The proposed routing algorithm effectively incorporates regional traffic information from multiple clusters, offering an informed perspective of traffic conditions to facilitate optimal path selection toward the destination. Each node generates a single bit of traffic information, which is only transmitted when the node is congested, thus effectively minimizing overhead. Finally, a path diversity parameter is employed to determine the most optimal route from the source to the destination, ensuring efficiency in the routing process. In the study conducted by Somisetty et al. (2022), it was emphasized that advancements in submicron technology have significantly increased the number of embedded intellectual properties (IPs) within System-on-Chip (SoC) architectures, thus enabling the development of multi-core SoCs. The conventional interconnection systems typically used in these SoCs lead to the creation of complex systems, often resulting in traversal delays. However, in the context of SoCs, Network-on-Chip (NoC) technology effectively mitigates the issues associated with conventional interconnections, providing a more efficient means of data transfer between cores. Within NoC architectures, there is an additional need for routing algorithms that not only minimize the number of hops but also avoid congested paths. To address this requirement, the study proposes a Negative-first, Congestion-aware Routing Algorithm with Fair Arbitration (CANF-FA), which ensures optimal path selection by considering both congestion levels and fair arbitration, thus enhancing overall routing efficiency.

The performance of a Network-on-Chip (NoC), similar to other types of networks, is largely influenced by the routing techniques implemented within it. The routing algorithm plays a crucial role in determining the path for data transmission from the source switch to the destination. An optimal routing algorithm should operate without causing issues such as deadlock, livelock, or starvation. Routing algorithms are generally classified into two categories: adaptive and deterministic. Deterministic routing algorithms consistently establish a fixed path for each specific source-destination pair. In such algorithms, the network's traffic conditions are not taken into account, meaning that the routing decisions are static and remain unchanged, regardless of network congestion or other dynamic factors. In contrast, adaptive routing algorithms are characterized by their ability to make routing decisions based on the dynamic traffic conditions of the network. These algorithms strive to redirect packets along alternative paths with lower congestion when the network experiences traffic overload. Both deterministic and adaptive routing algorithms present distinct advantages and disadvantages. Deterministic routing algorithms, due to their relatively simple routing logic, facilitate easier router design and result in lower routing delays in scenarios of light traffic congestion (Wang et al., 2023). However, when the network faces significant congestion, these algorithms experience a considerable increase in routing delays. This is primarily due to the inherent limitation of deterministic algorithms, which lack the flexibility to reroute packets through alternative paths when congestion is detected. In other words, deterministic routing algorithms lack the flexibility to utilize alternative paths to bypass congested links and regions. In contrast, adaptive routing algorithms dynamically adjust to the network's traffic conditions. These algorithms are capable of selecting alternative routes during periods of congestion, which results in reduced delays and, consequently, improved network throughput. However, under low-traffic conditions, adaptive algorithms typically incur higher delays compared to deterministic ones, owing to the increased complexity of their routing logic. Routing

algorithms in Network-on-Chip (NoC) are designed to facilitate data transmission between nodes within the network. Their primary objectives are to minimize delays, optimize energy consumption, and reduce interference, ensuring efficient network operation.

In Network-on-Chip (NoC) architectures, obtaining global information during the decision-making process for routing can have a profound effect on selecting the most efficient path to the destination. A major innovation of this research is the development of a communication subsystem specifically designed to gather global information. Additionally, the study introduces a routing algorithm that leverages both local and global information, which can significantly enhance the overall performance of the system, particularly in terms of throughput and delay.

PROPOSED METHOD

Global congestion information in a Network-on-Chip (NoC) encompasses comprehensive data reflecting the overall traffic status and network load across the chip. This information is obtained by aggregating and analyzing local congestion metrics collected from various network components, such as routers and links. Key parameters of global congestion information include the total number of queued packets, the average packet delay, and the overall bandwidth utilization. To enable efficient traffic management and optimal routing decisions, this information must be conveyed to the current node in real-time. By utilizing these data, routing algorithms can effectively assess network congestion and determine the most efficient paths for data transmission, thereby improving overall network performance.

In the proposed method, the time interval between two consecutive packet departures from a router's buffer is considered as the output delay, serving as an indicator of the router's congestion level. This delay is then propagated to neighboring nodes via dedicated interconnects. The dissemination of congestion information follows a structured approach: upon receiving congestion data from a neighboring router, the current router integrates it with its local congestion level using a weighted averaging scheme, where 40% of the final value is derived from the neighboring router's congestion and 60% from the local congestion level. The updated congestion information is subsequently forwarded to adjacent routers. This methodology enables congestion assessment along both the x and y dimensions, ensuring that traffic conditions are evaluated in alignment with the congestion levels of the respective directions. As a result, the routing algorithm gains a more comprehensive and dynamic global view of network congestion, allowing for more informed and efficient routing decisions. Figure 1 illustrates the designed communication structure used for global congestion information aggregation within the network.

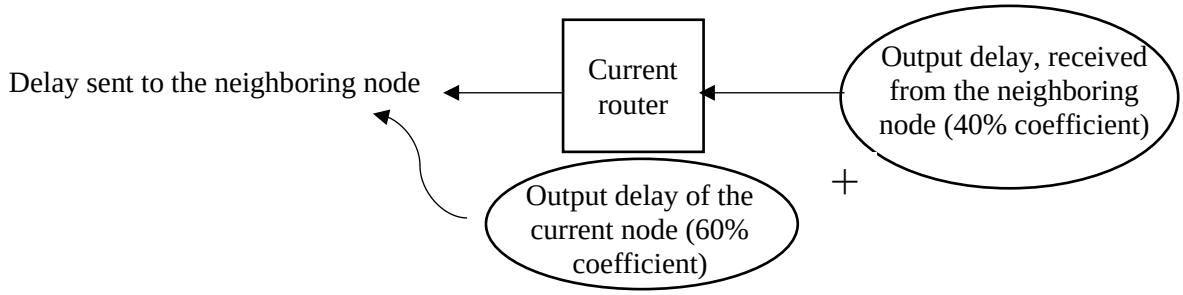

Figure 1: The proposed communication structure for aggregating global information.

In Figure 2, the flowchart of the proposed method is presented.

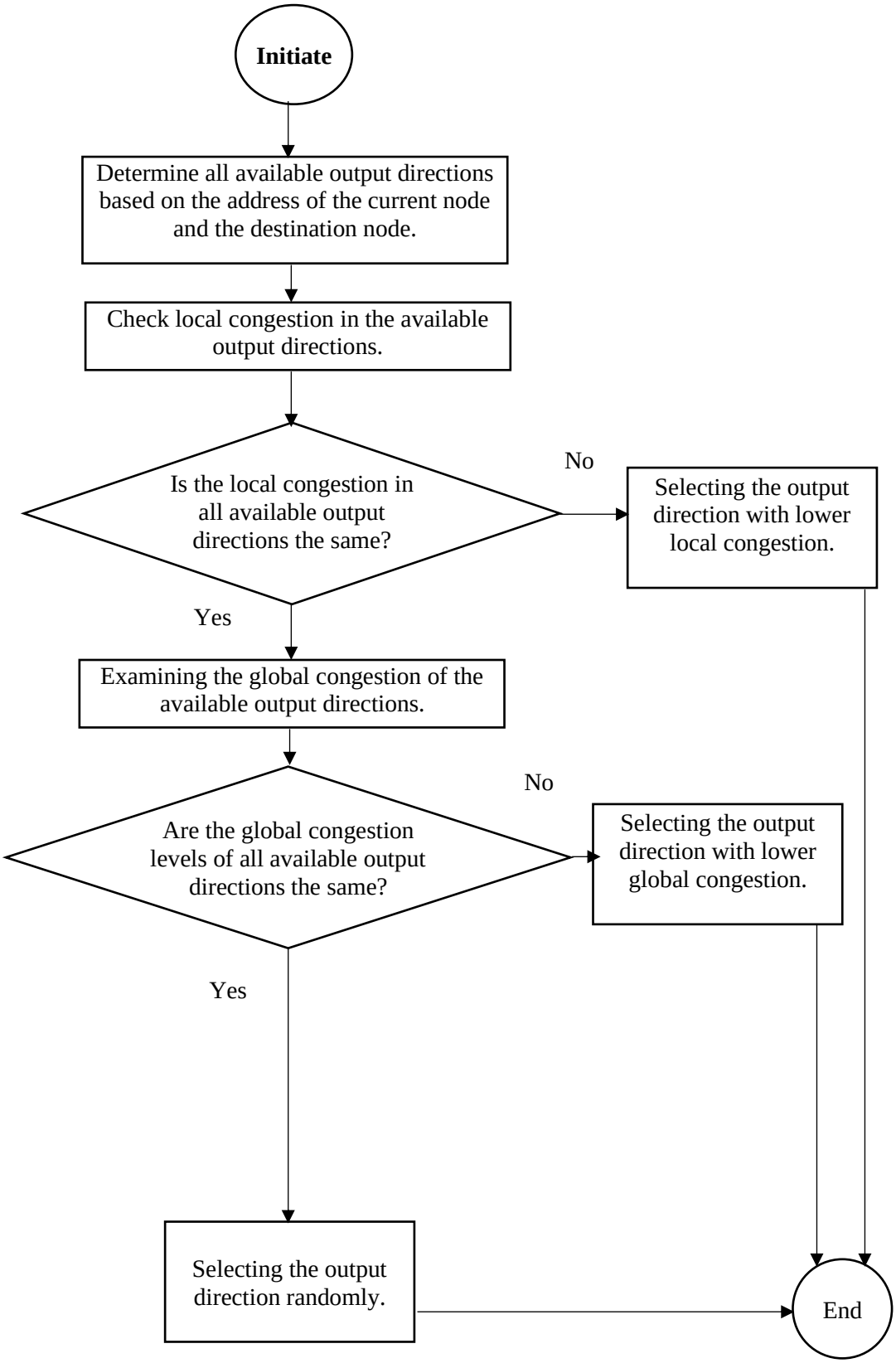

Figure 2: Flowchart of the proposed method.

As depicted in Figure 2:

The routing decision process begins by identifying the shortest possible paths based on the addresses of the current node and the destination node. This step determines the set of feasible output directions leading toward the destination. Once the potential output directions are identified, the next phase involves evaluating both local and global congestion levels to select

the optimal path. Initially, the local congestion of each available direction is assessed. If a particular direction exhibits lower local congestion compared to others, it is immediately selected as the output path. However, if multiple directions have the same local congestion level, the selection process proceeds to the next criterion—global congestion. At this stage, the global congestion values, which are propagated through dedicated interconnects and received by the current node, are analyzed. Among the remaining candidate directions, the one with the lowest global congestion is preferred. In scenarios where multiple directions exhibit identical local and global congestion levels, one direction is randomly selected to ensure load balancing and prevent routing bias. This approach enhances traffic distribution across the network and mitigates the risk of persistent congestion in specific paths.

Figure 3 illustrates the pseudocode of the proposed method, highlighting its programmability and practical implementation. This pseudocode provides a structured representation of the decision-making process in the routing algorithm and operates as follows:

First, the addresses of the current node and the destination node are retrieved to identify the shortest possible paths and determine the set of available output directions. Subsequently, both local and global congestion levels for these directions are assessed. If one of the available directions exhibits lower local congestion than the others, it is immediately selected as the output path. However, if multiple directions have the same local congestion level, the selection process proceeds to the next criterion—global congestion. The direction with the lowest global congestion is then chosen to optimize traffic flow. In cases where multiple directions share identical local and global congestion levels, one direction is selected randomly to ensure balanced traffic distribution and prevent deterministic routing patterns that could lead to network bottlenecks.

This approach enhances overall network performance by dynamically adapting to congestion conditions and ensuring efficient data transmission.

```
Proposed Algorithm Pseudocode
BEGIN
   // Step 1: Determine possible shortest paths and available output directions
   current_node_address = getCurrentNodeAddress ()
   destination_node_address = getDestinationNodeAddress ()

   possible_paths = findShortestPaths (current_node_address, destination_node_address)
   available_output_directions = determineOutputDirections(possible_paths)

   // Step 2: Assess local and global congestion for available output directions
   local_congestion = {}
   global_congestion = {}

   FOR each direction IN available_output_directions DO
      local_congestion[direction] = measureLocalCongestion(direction)
      global_congestion[direction] = measureGlobalCongestion(direction)
   END FOR

   // Step 3: Select the output direction with the least congestion
   min_local_congestion = MIN (local_congestion. values ())
   directions_with_min_local_congestion = FILTER (directions IN
available_output_directions WHERE local_congestion[directions] ==
min_local_congestion)

   IF LENGTH (directions_with_min_local_congestion) == 1 THEN
      selected direction = directions_with_min_local_congestion [0]
   ELSE
      min_local_congestion = MIN (global_congestion[direction] FOR direction IN
directions_with_min_local_congestion)
      directions_with_min_global_congestion = FILTER (directions IN
directions_with_min_local_congestion WHERE global_congestion[directions] ==
min_global_congestion)

      IF LENGTH (directions_with_min_global_congestion) == 1 THEN
         selected_direction = directions_with_min_global_congestion [0]
      ELSE
         // Step 4: If local and global congestion are the same, select a direction randomly
         selected_direction = SELECT_RANDOM (directions_with_min_global_congestion)
      END IF
   END IF

   // Output the selected direction
   outputDirection(selected_direction)
END
```

Figure 3: Pseudocode of the Proposed Method

To assess the performance of the proposed algorithm, this study employs two primary metrics: average packet delay and network throughput. These parameters are critically examined in the following sections to provide a comprehensive evaluation of the algorithm's effectiveness and efficiency in optimizing network performance.

The average delay is calculated using equation (1) and is typically expressed in terms of clock cycles.

$$L_{avg} = \frac{\sum_{i=1}^{p} L_i}{\mathrm{P}} \qquad (1)$$

In equation (1):

- P: The number of packets received within a specified period.
- L_i: The delay of the i-th packet.

Network throughput can be calculated using equation (2).

$$T = \frac{(Total\ Messages\ Completed)(Messages\ Length)}{(Number\ Of\ IP\ Blocks)(Time)} \quad (2)$$

In equation (2):

- Total Messages Completed: The number of messages successfully transmitted from the source to the destination within the network.
- Messages Length: The number of flits in a message.
- Number of IP Blocks: The number of processing cores.
- Time: The number of clock cycles between the generation of the first packet and the reception of the last packet in the network.

Table (1) outlines several key specifications and features of the simulation environment. For this simulation, the Noxim simulator has been utilized. The input parameters for this simulator include factors such as the routing algorithm, buffer size, network dimensions, and traffic type. Noxim was developed by the Computer Architecture Group at the University of Catania, Italy. The simulator is built using SystemC, which itself is based on the C++ programming language. Noxim includes a command-line interface that allows users to define, initialize, and input various parameters related to the Network-on-Chip (NoC) into the simulator. The simulation parameters are presented in Table (1).

Table 1: Parameters of the Simulation

Network dimensions	2D 6×6 Mesh Network
Routing algorithm	Odd/Even
Number of simulation cycles	100,000 cycles
Number of steady-state cycles	2,000 cycles
Traffic types under evaluation	Random, transpose
Buffer size	4 flits
Packet length	4 flits

FINDINGS

1. **Simulation Results under Uniform Traffic Pattern**

The graphs illustrating the average packet delay and throughput indicate a noticeable improvement in network performance with the proposed algorithm. These graphs highlight the relationship between the packet injection rate (Packet/cycle/IP) and the average packet delay (cycles) across the three methods being compared. At lower injection rates (approximately 0.012 to 0.017 Packet/cycle/IP), all three methods exhibit relatively low delays and demonstrate comparable performance, with packet delays ranging from 10 to 20 cycles. However, as the injection rate surpasses approximately 0.017, the differences between the methods become more evident. The approach proposed by Manzoor et al. shows a sharper increase in packet delay, while the method by Somisetty et al. also experiences an increase in

delay, though not as significantly as the approach by Manzoor et al. The proposed method exhibits a substantial improvement over the other two methods, with a lower delay. At injection rates exceeding 0.0185, the approach by Manzoor et al. experiences a sharp increase in delay, reaching approximately 360 cycles, whereas the method by Somisetty et al. also faces a significant delay increase, although less severe, reaching about 260 cycles. Despite this, the proposed method continues to show superior performance, maintaining a lower delay, though a slight increase in delay is also observed in this approach. In general, the proposed method outperforms the other two approaches in terms of average packet delay at all injection rates, making it a promising option for enhancing the performance of on-chip networks. In contrast, the method by Manzoor et al. shows the weakest performance at higher injection rates, while the method by Somisetty et al., though experiencing increased delays at higher injection rates, still performs better than the approach by Manzoor et al.

Table 2: Results of average packet delay in the uniform traffic pattern.

Injection rates	Manzoor et al	Somisetty et al	Proposed
0.012	19.8069	19.1046	18.4053
0.013	21.6125	20.6242	19.8353
0.014	23.6088	22.3987	21.3458
0.015	26.1599	24.9369	23.0514
0.016	30.3653	28.0987	25.3503
0.017	38.0766	33.9708	28.7179
0.018	61.7555	65.3814	33.6834
0.0186	287.0935	128.2571	67.7868
0.0189	880.8751	282.1423	233.476

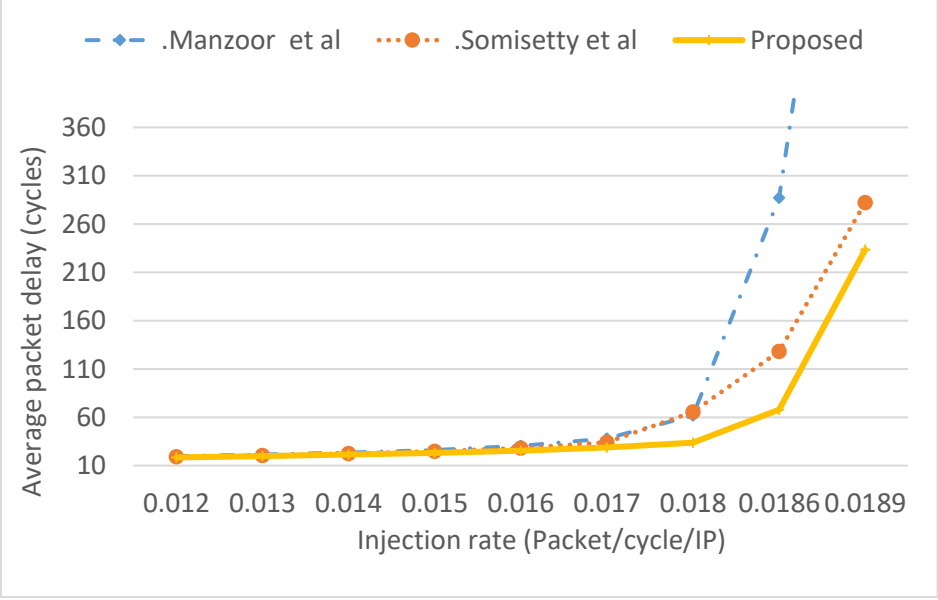

Figure 4: Comparison of average packet delay in the uniform traffic pattern.

Figure 5 illustrates the analysis and evaluation of network throughput for the three algorithms under comparison. The graph demonstrates the relationship between the packet injection rate (Packet/cycle/IP) and network throughput (Flits/Cycles/IP) across three distinct methods. At low injection rates (ranging from approximately 0.012 to 0.017 Packet/cycle/IP), all three methods exhibit comparable throughput. As the injection rate increases, the network throughput rises linearly. At moderate injection rates (around 0.018), the network throughput reaches its peak, with the proposed method and the approach by Somisetty et al. achieving

similar performance, while the method by Manzoor et al. slightly underperforms. After this point, network throughput begins to decline for all three methods, with the rate of decrease being more pronounced in the approach by Manzoor et al. The method proposed by Somisetty et al. also experiences a reduction in throughput, though the decrease is less severe compared to Manzoor et al.'s method. The proposed method outperforms the others by exhibiting the least reduction in throughput, maintaining higher throughput at elevated injection rates compared to both alternative methods. In summary, the proposed method consistently achieves superior throughput across all injection rates, demonstrating a smaller decline at higher rates than the other two approaches. Therefore, this method is recommended for improving throughput in Network-on-Chip systems. The numerical data corresponding to Figure 5 is presented in Table 3.

Table 3: Results of network throughput under the uniform traffic pattern.

Injection rates	Manzoor et al	Somisetty et al	Proposed
0.09578	0.09667	0.09612	0.09578
0.10385	0.10427	0.10361	0.10385
0.11194	0.11176	0.11213	0.11194
0.12014	0.1201795	0.11976	0.12014
0.12834	0.128026	0.12795	0.12834
0.13533	0.1358235	0.13599	0.13533
0.14389	0.143167	0.143701	0.14389
0.146832	0.145207	0.14645	0.146832
0.146684	0.1422965	0.148951	0.146684
0.146605	0.1412246	0.150965	0.146605
0.145018	0.139288	0.151346	0.145018

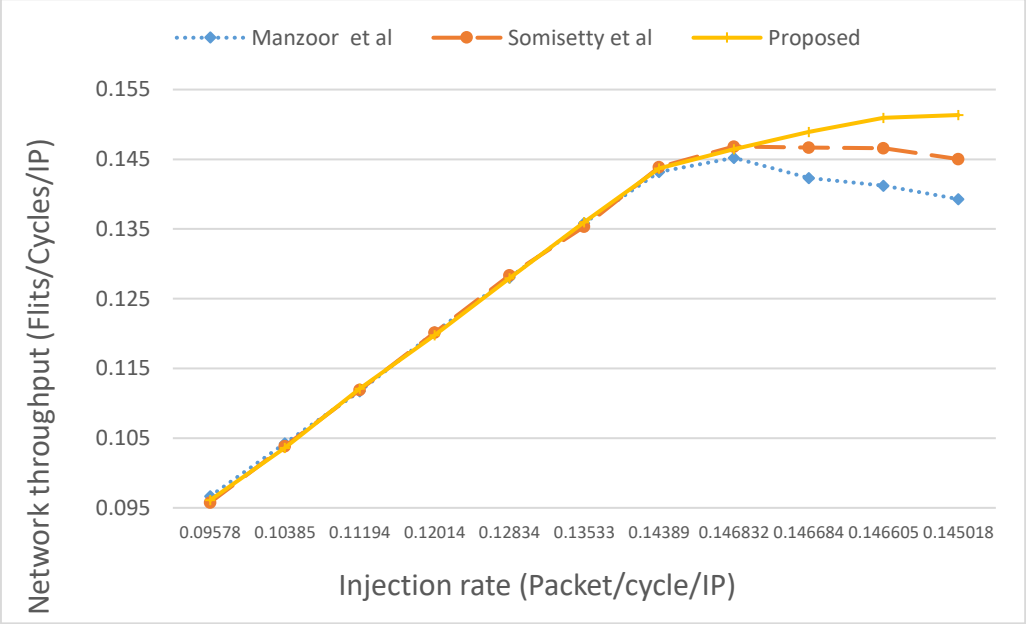

Figure 5: Network throughput in the uniform traffic pattern.

2. Simulation Results under the Transitional Traffic Pattern

The simulation results for the parameters of average packet delay and network throughput under the transitional traffic pattern are presented below. Upon analyzing the graph, it becomes evident that, within the lower injection rate range, all three routing methods exhibit relatively

low delays and show similar performance. As the injection rate increases to approximately 0.017 Packet/cycle/IP, the disparities between these methods become more apparent. In this range, the approach proposed by Manzoor et al. demonstrates a more pronounced increase in packet delay compared to the other methods, while the method by Somisetty et al. also experiences a delay increase, but to a lesser extent than Manzoor et al.'s approach. However, the proposed method outperforms the other two approaches up to an injection rate of around 0.018. Once the injection rate exceeds 0.018, the delay in the proposed method increases sharply, with a significant rise in delay observed at higher injection rates. As a result, although the proposed method delivers a noticeable performance improvement in terms of packet delay at low and moderate injection rates, it requires further optimization to maintain its efficiency at high injection rates. The numerical details corresponding to Figure 6 are presented in Table 4.

Table 4: Average packet delay results under the transitional traffic pattern.

Injection rates	Manzoor et al	Somisetty et al	Proposed
0.012	17.18015	18.60903333	16.38055
0.013	18.66683333	20.34273333	19.3402
0.014	20.94775	22.9576	20.25323333
0.015	23.59498333	27.58126667	20.87123333
0.016	28.66356667	33.03213333	24.73785
0.017	35.67446667	53.28673333	31.75046667
0.018	52.41933333	155.04715	44.32308333
0.019	155.3526167	514.1991667	94.78973333
0.02	455.6266667	914.2981667	153.8378

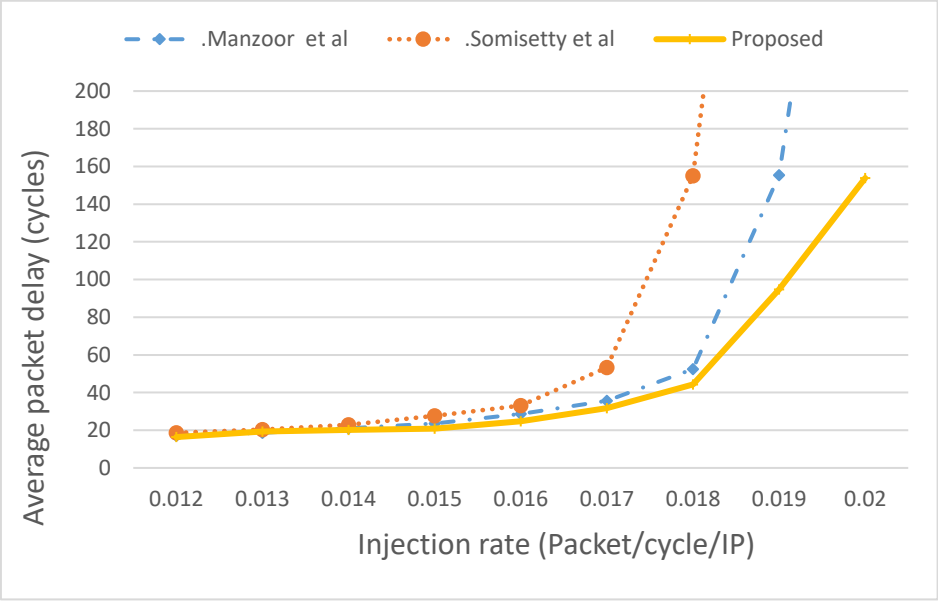

Figure 6: Average packet delay chart under the transitional traffic pattern.

Figure 7 illustrates the network throughput results for the algorithms being compared, alongside the proposed algorithm.

Table 5: Network throughput results under the transitional traffic pattern.

Injection rates	Manzoor et al	Somisetty et al	Proposed
0.012	0.095885383	0.09632595	0.0960178
0.013	0.104106667	0.103825667	0.104296833
0.014	0.111897333	0.112108333	0.112232
0.015	0.1196445	0.119928167	0.119851333
0.016	0.128243333	0.128090667	0.127631833
0.017	0.136366667	0.136232833	0.1366885
0.018	0.1441205	0.143739167	0.144438333
0.019	0.149843333	0.146317833	0.152170167
0.02	0.149243333	0.145817833	0.152570167

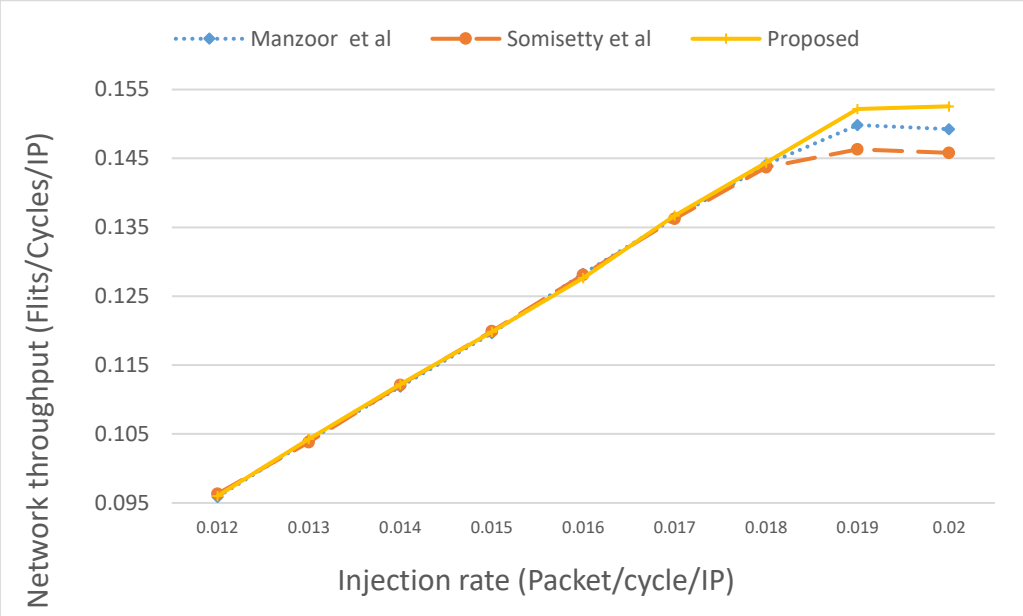

Figure 7: Network throughput chart under the transitional traffic pattern.

As depicted in Figure 7, the proposed algorithm achieves superior network throughput compared to the other methods, thereby contributing to a significant enhancement in overall network performance.

CONCLUSION

In this research, a congestion-aware algorithm was proposed, leveraging both local and global congestion data. Built upon the suggested information-gathering structure, the algorithm effectively avoids congested regions within the network, thereby improving overall network performance. The results of simulations and evaluations conducted under two traffic patterns—uniform and random—highlighted the superior performance of the proposed algorithm when compared to alternative approaches. Significant improvements were observed in the network's performance, particularly concerning average packet delay and overall throughput. As part of future work, it is recommended to explore the integration of artificial intelligence algorithms for predicting congestion and errors within the network.

REFERENCES

1. Fang., J, Mao, Y., Cai, M., Zhao, L.A., Chen, H., Xiang, W., 2022, STTAR: A Traffic-and Thermal-Aware Adaptive Routing for 3D Network-on-Chip Systems. Computers, Materials & Continua.;72:5531-45.
2. Gogoi, A., Ghoshal, B., Manna, K., 2023, Fault-aware routing approach for mesh-based Network-on-Chip architecture. Integration. Jun 22.
3. Joshi, B., Thakur, M.K., 2023, Genetic algorithm-and cuckoo search algorithm-based routing optimizations in network-on-chip. Arabian Journal for Science and Engineering. Aug;48(8):9635-44.
4. Lee, Y.S., Kim, Y.W., Han, T.H., 2022, Mrcn: Throughput-oriented multicast routing for customized network-on-chips. IEEE Transactions on Parallel and Distributed Systems. Oct 26;34(1):163-79.
5. Manzoor, M., Mir, R.N., 2022, PAAD (Partially adaptive and deterministic routing): A deadlock free congestion aware hybrid routing for 2D mesh network-on-chips. Microprocessors and Microsystems. Jul 1; 92:104551.
6. Narayana, S.Y., Mandal, S.K, Ayoub, R., Kishinevsky, M., Ogras, U.Y., 2023, A Lightweight Congestion Control Technique for NoCs with Deflection Routing. In2023 Design, Automation & Test in Europe Conference & Exhibition (DATE) Apr 17 (pp. 1-2). IEEE.
7. Somisetty, R., Karthik, V.S., Srujan, M.R., Vinodhini, M., 2022, Congestion aware negative first routing with fair arbitration for network on chip. In2022 6th International Conference on Computing Methodologies and Communication (ICCMC) Mar 29 (pp. 215-220). IEEE.
8. Vazifedunn, S., Reza, A., Reshadi, M., 2023, Low-cost regional-based congestion-aware routing algorithm for 2D mesh NoC. International Journal of Communication Systems. Jan 10;36(1): e5360.
9. Wang, S., Zhang, X., Dong, D., Li, C., Wang, Z., Zhang, Z., 2023, LARE: A Linear Approximate Reinforcement Learning Based Adaptive Routing for Network-on-Chips. In2023 IEEE International Symposium on Circuits and Systems (ISCAS) May 21 (pp. 1-5). IEEE.
10. Zhou, W., Ouyang, Y., Xu, D., Huang, Z., Liang, H., Wen, X., 2023, Energy-efficient multiple network-on-chip architecture with bandwidth expansion. IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, 31(4), pp.442-455.