Review, Analysis, and Implementation of Path Selection Strategies for 2D NoCs

RAJENDRA SINGH¹, MANOJ KUMAR BOHRA¹, (Senior Member, IEEE), PRASHANT HEMRAJANI¹, ANSHUMAN KALLA², DEVERSHI PALLAVI BHATT³, NITIN PUROHIT⁴, AND MASOUD DANESHTALAB⁵,⁶

¹Department of Computer and Communication Engineering, Manipal University Jaipur, Jaipur 303007, India
²Chhotubhai Gopalbhai Patel Institute of Technology (CGPIT), Uka Tarsadia University, Bardoli, Gujarat 394620, India
³Department of Computer Application, Manipal University Jaipur, Jaipur 303007, India
⁴Department of Computer Science, Kebri Dehar University, Kebri Dehar 3060, Ethiopia
⁵Mälardalen University, 722 20 Västerås, Sweden
⁶Tallinn University of Technology, 12616 Tallinn, Estonia

Corresponding authors: Nitin Purohit (nitinpurohit111@kdu.edu.et) and Manoj Kumar Bohra (bohra.manoj1980@gmail.com)

ABSTRACT Recent advances in very-large-scale integration (VLSI) technologies have offered the capability of integrating thousands of processing elements onto a single silicon microchip. Multiprocessor systems-on-chips (MPSoCs) are the latest creation of this technology evolution. As an interconnection network, Network-on-Chip (NoC) has emerged as a scalable and promising solution for MPSoCs to achieve high performance. In NoCs, a routing algorithm is a critical part of a router and provides a path for a packet toward its destination. Every routing algorithm should exhibit two characteristics. First, the route selection function should provide enough degree of adaptiveness to avoid network congestion. Second, it should not offer stale information on network congestion status to the neighboring routers. Many researchers have investigated network congestion and proposed techniques to control/avoid congestion. Such congestion avoidance-based algorithms significantly improve NoC performance. However, they may result in hardware overhead for side network implementation to collect congestion status. This paper reviews various output selection strategies used by routing algorithms to route a packet on a less congested network region. It also classifies them based on techniques adopted to handle and propagate congestion information. Additionally, this article provides the implementation and analysis details of state-of-art selection methods.

INDEX TERMS Network-on-chip, routing algorithm, congestion, selection methods, traffic distribution.

I. INTRODUCTION

The density of on-chip transistors has dramatically increased with Moore’s law, enabling the consolidation of thousands of cores within a single die. These cores may be processors, cache banks, or other heterogeneous resources which collectively form System-On-Chip (SoC). Network-On-Chip (NoC) has augmented as a reliable solution to the traditional bus-based on-chip interconnection and addresses performance, scalability, and acceleration issues of the real-time embedded system applications [1], [2], [3]. NoC based solutions exhibit higher communication bandwidth, modular architecture, and scalability. It can provide better congestion control and fault-tolerant capability than traditional infrastructures [4]. NoC architectures flatter the basic communication structure for Chip-MultiProcessors (CMPs).

NoC performance depends on different network characteristics, like topology, switching method, routing algorithm, output-selection method, flow-control techniques, etc.

- Topology: In NoC, the topologies define the interconnection of routers in the network. Many network topologies have been implemented as a soul for on-chip interconnection networks, such as mesh, torus, spin, flattened butterfly, and spidergon networks [5], [6], [7], [8], [9], [10]. Among them, 2D meshes are widely used for their simplicity and flexibility [5], [6], [7], [8], [9], [10], [11], [12], [13], [14].
- Switching Techniques: In NoC, switching techniques define how and when packets will be allocated network resources during their travels. Different NoC imple-
Routing methods are classified as deterministic and adaptive. Deterministic routing methods do not consider the current network status to forward packets. These methods always compute the same route for a given pair of source and destination nodes. For example, dimension-order routing (say XY) routes a packet in one dimension (X) first to reduce offset to zero and then route to another dimension (Y). However, in adaptive routing techniques, a packet can choose any available route from the current node to the destination node.

The functionality of a routing method can be divided into two parts: the route computation and selection functions, as shown in Fig. 1. The route computation function provides the output directions that may be one or two for a minimal path in a 2D Mesh NoC. However, if two output directions are generated by routing function, this will introduce a decision problem for selecting one direction out of these two [18]. This indecision situation has resulted in the development of selection strategies that aim to find a direction toward a less congested network region.

NoCs have to face non-uniform traffic distribution in the network resulting from different factors, such as routing biases, topological artifacts, long-range dependencies, traffic scenarios, link failures, etc. Such non-uniform traffic has a vital impact on the performance of NoC [19], [20], [21]. Such traffic distribution results in ill-congested regions in the network, increasing network delay and lowering the NoC performance.

To control/avoid congestion in the network, researchers have proposed solutions using dedicated wire networks and packet header-free bits to assist in the timely collection and transmission of congestion information. These implementations introduce additional problems, as dedicated wires put extra hardware overhead on the system, and the packet header bit introduces information staleness, which may result in the less effective selection of the output channel. Therefore, the adoption of a selection strategy has a major impact on communication performance.

Significant work has been published on the improvement of NoC routing that reduces network latency and increases NoC throughput. To eliminate congestion, they use local and regional information with justified weight. However, we believe there are still possibilities to improve NoC performance by reducing congestion in the network. We can propagate fresh congestion information between routers by precise prediction of traffic and congestion status using machine learning techniques.

### A. MOTIVATION

Table 2 summarizes the several routing and output-selection techniques related survey paper for NoC. Bjerregaard and Mahadevan [22] discuss the research activity in NoC with its basic concept and highlight the necessity for a modular design approach in NoC for better utilization of resources. Agarwal et al. [17] have discussed NoC architecture and hardware issues and application mapping strategies for it. The classification of these application mapping strategies has been provided by Sahu and Chattopadhyay [23], which classify in static and dynamic approaches and show their performance comparison. Abbas et al. [24] present a review of mapping strategies with their fault tolerance capabilities. They highlighted the need to target mapping strategies to solve the fault in the network. In [25], [26], [27], [29], and [30], authors provide a survey on routing algorithms and their classification. Gabis and Koudil [25] classify based on protocol uses, Wu et al. [26] classify based on path diversity, Zulkifli et al. [27] compare adaptive routing, Ahmad and Sethi [29] review routing algorithms and Kaleem and Isnin [30] classify based on path diversity with criteria such as thermal aware, aging aware, power efficiency, fault aware and resilience. Reference [28] present survey on congestion control mechanism in WiNoC.

Congestion control is a widely studied area in NoC to improve average delay and throughput as it directly relates to the performance of NoC. Many surveys were submitted for NoCs that target different aspects of NoC performance, some providing path diversity-based comparisons and others discussing congestion control mechanisms for NoC. To the best of our knowledge, there is still a gap in the survey of congestion-based routing algorithms which focuses on congestion information prediction, its propagation and their...
TABLE 1. List of important acronyms.

| Acronyms | Definitions |
|----------|-------------|
| ACDC | Accuracy and Congestion aware Dynamic traffic Control |
| ACO-CAR | ACO based Cascaded Adaptive Routing |
| AQFE | Adaptive Q routing technique with Full Echo extension |
| BARP | Balanced Adaptive Routing Protocol |
| CATRA | Congestion Aware Trapezoid based Routing Algorithm |
| CHARM | Congestion-aware Highly Adaptive Routing Method |
| DAR | Destination based Adaptive Routing |
| DPLPR | Dynamic Programming based Lifetime and Performance aware Routing |
| DPXY | Distance Prediction XY |
| DyXY | Dynamic XY |
| e-SBTR | Express virtual channel -State Based Transmission Line Routing |
| FACARS | Fully Adaptive Congestion Aware Routing Scheme |
| GCA | Global Congestion Aware |
| HDN | Hierarchical Deflection Routing |
| IIRA | Heuristic based Routing Algorithm |
| LBRA | Lattice Based Routing Algorithm |
| LTCA | Load Balanced time based Congestion Aware |
| Mini ESPADA | Mini EScape Paths with Dynamic channel Acquisition |
| NIR | Network Information Region |
| NoP | Neighbor-on-Path |
| PCQ | Probabilistic Congestion based Q-routing |
| PEAPAR | Power Efficient Partially Adaptive Routing |
| PW Odd Even | Preemptive Waiting Odd Even |
| QCA DCT | Quantitative Congestion Avoidance - Directly the local Congestion Status information |
| RTM | Repetitive Turn Model |
| SoC | System-on-Chip |
| SRM | Spike Response Model |
| URCA | UniBiased Regional Congestion Aware |
| σLBD | σ-bits Logic Based Distributed Routing |

| Acronyms | Definitions |
|----------|-------------|
| ACO | Ant Colony Optimization |
| ALA | Adaptive Look Ahead |
| AQRERM | Adaptive Q routing with Random Echo and Route-Memory |
| CACBR | Congestion Aware Cluster Buffer base Routing algorithm |
| CFPA | Congestion-aware, Fault-tolerant and Process-variation aware Adaptive routing algorithm |
| CrQ | Credence based Q-routing |
| DBAR | Destination Based Adaptive Routing |
| DPLR | Dynamic Programming based Lifetime aware Routing |
| DyAD | Dynamic Adaptive Deterministic switching |
| DyXYX | Dynamic XY XY |
| E0mesh | Even Odd Mesh |
| FLR | Fuzzy Logic Routing |
| HABR | History aware Adaptive Routing |
| HOEQ | Hybrid Odd Even Q-routing |
| LAXY | Location based Aging resilient Xy Yx |
| LCCDM | Low Cost Congestion Detection Mechanism |
| MICRA | Minimal Congestion Aware Routing Algorithm |
| MWPF | Modified Whole Packet Forwarding |
| NoC | Network-on-Chip |
| PAR | Partially Adaptive Routing |
| PreNoC | Predictive Routing for NoC |
| QCA | Quantitative Congestion Avoidance |
| RCA | Regional Congestion Aware |
| SNN | Spiking Neural Network |
| SoRA | Sub-optimal Routing Algorithm |
| TA | Traffic Allocation |
| WAR | Workload Aware Routing |

This analysis. This has motivated us to work on this paper so as to provide congestion controlling and avoiding strategies based classification of routing and selection algorithms for NoC.

B. PAPER CONTRIBUTION

This paper targets the various known solution provided by research community for controlling congestion situations in NoC. The contributions of this paper are listed below.

- This article provides a broad survey of routing and selection algorithms. It discusses recent and state-of-the-art routing and selection techniques for 2D Mesh topology considering various related parameters.
- It presents different classifications of routing and selection algorithms on the basis of congestion avoidance and control strategies. It discusses congestion type and its effect on selection of output direction. This research provides better understanding of congestion situations and their handling in the network.
- We provide future directions by discussing the effect of congestion through the result & analysis section. This section addresses the need for machine learning techniques to predict precise congestion information.

C. ORGANIZATION OF THE PAPER

In this paper, the study focuses on investigating the routing and selection algorithms for 2D mesh NoC. We include all recently proposed algorithms and some soul routing techniques in our research. Further organization of the paper is as follows; Section II discusses the challenges faced by various routing and selection strategies for achieving higher performance. Section III provides a classification of routing and selection strategies proposed in the past. Section IV presents information about simulator and evaluation metrics of previously published state-of-art. Section V presents the result and analysis of various previously published work on congestion, and Section VI concludes the paper and provides future direction.
TABLE 2. Summary of survey papers in NoC.

| Reference               | Year | Key Contribution                                                                 | Focused Area                      |
|-------------------------|------|----------------------------------------------------------------------------------|-----------------------------------|
| Bjerregaard and Mahadevan [22] | 2006 | • Showcased complete design abstractions range.                                | NoC Architecture design           |
|                         |      | • Figured out general trends and issues in NoC Architecture Communication.       |                                   |
|                         |      | • Presented NoC implementation case studies.                                    |                                   |
| Agarwal et al. [23]     | 2009 | • Discussed NoC hardware and architecture-related issues.                       | NoC architecture and application mapping strategies |
|                         |      | • Highlighted the need for an integrated modeling approach for NoC architecture. |                                   |
| Sabu and Chattopadhyay [24] | 2013 | • Presented recent application mapping strategies for NoC.                     | NoC application mapping strategies |
|                         |      | • Presented classification of application mapping as static and dynamic application mapping approach. |                                   |
|                         |      | • Presented comparison between various static mapping techniques.              |                                   |
|                         |      | • Presented some custom test cases with 64 and 128 cores.                       |                                   |
| Abbas et al. [25]       | 2014 | • Provided review of mapping technique that support fault tolerance.            | NoC application mapping and fault tolerance technique |
|                         |      | • Highlighted that 3D topology creates issues for mapping reliability and performance. |                                   |
|                         |      | • Suggested multi-objective optimization approaches for fault tolerant application mapping. |                                   |
| Gabis and Koudil [26]   | 2016 | • Presented classification of NoC routing algorithm on the objectives (i.e. congestion, failures, deadlock and livelock) for which they are used. | Objectives of routing algorithms in NoC |
|                         |      | • Discussed the principles, features and techniques used for achieving algorithms objective, and their impact on performance. |                                   |
| Wu et al. [27]          | 2016 | • Presented recent research trend for NoC routing.                             | Routing algorithms in NoC         |
|                         |      | • Discussed the routing algorithms as oblivious, adaptive and half-adaptive.     |                                   |
| Zulkefli et al. [28]    | 2017 | • Compared NoC adaptive routing algorithms.                                     | Adaptive routing algorithms in NoC |
|                         |      | • Highlighted that researchers focused on wormhole flow control and turnaround algorithm for designing router. |                                   |
| Rad et al. [29]         | 2020 | • Discussed congestion problems and challenges in WiNoC (Wireless NoC).        | Congestion control mechanism in WiNoC |
|                         |      | • Presented the survey of congestion control mechanism.                         |                                   |
|                         |      | • Presented the comparison of state-of-the-art congestion control mechanisms in WiNoCs. |                                   |
| Ahmad and Sethi [30]    | 2020 | • Presented comparison of routing algorithms on various parameters such as routing type, latency, throughput, power consumption etc. | Routing algorithms in NoC         |
| Keleem and Isnin [31]   | 2021 | • Discussed the routing algorithm and mesh topology concepts.                   | Oblivious and adaptive routing algorithms in NoC |
|                         |      | • Presented the survey of oblivious and adaptive routing algorithms with criteria such as thermal aware, ageing aware, power efficiency, fault aware and resilience. |                                   |
| Our Work                | 2022 | • Discuss Congestion control mechanism for NoC.                                 | Routing and selection algorithms for congestion control in NoC |
|                         |      | • Provide congestion avoiding and controlling based classification of routing and selection algorithms. |                                   |
|                         |      | • Discuss various congestion information and their effect.                      |                                   |
|                         |      | • Perform simulation of various state-of-the-art algorithms and discuss their result with effect of various congestion information on performance of NoC. |                                   |

II. CHALLENGES IN ROUTING ALGORITHMS

To achieve high performance, routing algorithms should exhibit some basic requirements. First, the routing algorithm should provide adaptivity so that it has option to route packets toward less congested region. Second, it should be able to handle deadlock and live-lock situation. On the other hand, selection methods should be designed in such a way that these should be able to predict congestion situations precisely based on fresh congestion information. These should also handle faulty link situations. These challenges of routing and selection methods are discussed in detail as below.

A. DEADLOCK

The routing strategy’s primary challenge is to provide a deadlock-free network for the packets. A deadlock occurs in NoC when some packets are indefinitely blocked in a circular way. In deadlock, they wait for resources that are held by another packet and hold resources requested by other packets, thus creating cyclic dependency [31], [32]. An example of a deadlock situation is shown in Fig. 2, where router D to A, A to B, B to C, and C to D are connected with channels 1, 2, 3, and 4, respectively. At router A, packet 2, coming from north, holds channel 2 in the south direction and waits for channel 3 in the east direction of router B. However, at router B, channel 3 is held by packet 3, and packet 3 is also waiting for channel 4 in the north direction of router C. Similarly, Packets 4 and 1 hold channels 4 and 1 respectively, and wait for channels 1 and 2, respectively. Holding some channels and waiting for another channel in a cycle creates a deadlock situation in the network. Three strategies have been used to deal with deadlock situations: deadlock prevention, deadlock avoidance, and deadlock recovery. In deadlock prevention, resources are allocated (by blocking circular dependency) to packets so that deadlock will never occur. In this mechanism,
packets reserves all required channels before packet injection by the source node, thus inefficient. In deadlock avoidance, resources are allocated for targeting to avoid any condition that leads to a deadlock situation. In deadlock recovery, initially, no action is taken, whereas a mechanism has been provided to detect deadlock and then take action to eliminate deadlocks. According to [33], a network is free from deadlock until a cyclic dependency arises in Channel-Dependency-Graph (CDG). Two strategies have been adopted to break such cyclic dependencies. One is to allow only selected turns and prohibit other ones implemented by turn-based routing [31], [34]. The second is to use virtual channels where it switches between them on certain conditions to avoid deadlock [33], [35], [36], [37].

B. LIVELOCK
A livelock situation occurs in NoC when a packet is trapped in a circular path and running endlessly. This situation generally arises when resources necessitated for reaching the destination are occupied by some other packets, and the trapped packet may acquire and free the resources in iteration without any progress. An example of a livelock situation is shown in Fig. 3, where router 0 is the source and router 15 is the destination. A packet moves from 0 to 8 with a standard routing algorithm. The link from 8 to 12 is congested; thus, the packet is routed to 9. Links from 9 to 13 and 10 are congested; thus packet is misrouted to 5. At 5, it is also misrouted to 4 because of congestion. This misrouting of the packet makes the packet move in a cyclic path forever in 4, 8, 9, and 5, thus creating a livelock situation. Many authors have developed various strategies to take out the packet from livelock [32], [35], [36], [37].

C. STARVATION
A starvation situation occurs in NoC when a resource requested by a packet is never assigned. Such resources are continuously assigned to other packets resulting from the vicious allocation policy. A correct resource allocation policy can avoid starvation situations [32]. Since starvation problems arise in resource allocation policy, thus routing strategies do not affect it.

D. CONGESTION
Congestion is a state that arises when the flow of packets is concentrated on some paths that create saturated regions. Due to these saturated regions, the network performance starts to deteriorate. To avoid such saturation situations and improve network performance, several congestion-aware routing algorithms have been proposed that try to distribute packets throughout the network evenly. There are a variety of measures adopted to measure congestion. In [38], authors use stress value, and in [39] authors use buffer load information, and some authors use network pressure as a metric. Also, in [4], they give importance to packets from congested areas to de-saturate those areas.

E. FAULT
In NoC, a fault may appear because of logical errors, physical failures, or misbehavior of the network components. These faults generally result from changing logical values, fluctuation in current & voltage, or error signals in packets & routers. Such fault blocks the normal functioning of routers, and as a result, it degrades NoC performance [40], [41]. Thus, for proper and reliable functioning of NoC, efficient fault-tolerance algorithms are required.

III. CLASSIFICATION OF ROUTING ALGORITHMS
The aim of routing algorithms is to find the optimal route in NoC to reduce average delay and improve network throughput. To improve NoC performance, researchers have proposed many research articles that target the even
distribution of workload, lowering congestion, and avoiding faulty nodes. In this paper, we classify routing algorithms into four categories:

- workload distribution,
- congestion control,
- both workload distribution & congestion control,
- congestion control & fault tolerance

This classification is based on the process adopted by the algorithms to distribute workload that can depend on or be independent of the congestion status of the network. Table 3 summarizes various methods for solving network congestion related issues in NoC.

**TABLE 3. Categorization of routing and selection algorithms.**

| Routing and selection Algorithm | Workload Distribution | Congestion Control | Fault Tolerance |
|---------------------------------|-----------------------|--------------------|-----------------|
| [44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65] | ✓                     | x                  | ✓               |
| [18, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98] | x                     | ✓                  | ✓               |
| [99, 100, 101, 102, 103]          | ✓                     | ✓                  |                 |
| [104, 105, 106]                  | ✓                     | ✓                  | ✓               |

### A. WORKLOAD DISTRIBUTION

One possible solution to address the problem of network congestion is to distribute the workload evenly to all available paths. The increasing workload on certain parts of the network creates a congestion situation that affects the performance of NoC. Many algorithms (as shown in Table 4) have been proposed to handle such congestion situations that distribute the traffic load evenly in the network.

The turn model based algorithms can be the one way to distribute workload evenly, Chiu [42] present Odd-Even routing that applies restrictions on odd and even columns of the mesh network. It shows significant improvement in performance over existing turn model-based routing for various source-destination pairs. Sadrosadati et al. [48] propose Preemptive-Waiting for Odd-Even turn to improve adaptivity of Odd-Even routing. They separate turn prohibiting as direct and indirect, where direct prohibiting uses a policy to wait only for a definite period, after that, it starts routing using Odd-Even, and indirect prohibiting uses Odd-Even eternal. This direct and indirect prohibiting helps in increasing adaptivity and avoid deadlock situations. Another work on increasing adaptivity has been presented by Nasir and Zarandi [49]; they proposed MWPF based on Whole-Packet-Forwarding (WPF) that moderates WPF’s negative impact on packet latency by improving full output buffer problem.

Several turn model based techniques use minimal or non-minimal paths by applying restrictions on turns. Driving down these turn restrictions (meaning allowing more turns) can significantly improve the diversity of the algorithm. Such an approach (CHARM) is proposed by Kumar et al. [44] that lowers the routing turn restrictions. Another technique with the name “Nue” was presented by Domke et al. [45] that avoids deadlock implicitly whenever a path is identified by routing instead of separately handling deadlock and route path. Nue routing depends on the destination and has been designed to use the search-graph-based algorithm for changeable NoC topology. Jalili et al. [50] present Power-Efficient-Partially-Adaptive-Routing that uses partial adaptivity to achieve almost fully adaptivity. They divide all nodes between source and destination into positive and negative sets. Using two phase routing (first in positive and in negative direction) consecutively, algorithm achieves almost fully adaptive routing without using virtual channels and their complex implementation. Wang and Valencia [51] has proposed Traffic Allocation (TA) routing that monitors different direction traffic flow information of a router and maintains registers to store each direction information. This strategy helps distribute traffic more evenly and does not involve additional communication overhead to get neighbor buffer status. Tang et al. [52] presents Repetitive-Turn-Model (RTM) that utilize routing approaches with prohibited turn along both rows and columns and choose algorithms with lesser routing pressure in comparison to the Odd-Even algorithm. RTM assembles routing algorithms based on the distribution of prohibited turns from the smaller parts of the given network to the whole network. It mainly focuses on prohibiting certain turns along with columns and rows, unlike repeatedly prohibiting the same turns. Fusella and Cilardo [53] propose Lattice-Based-Routing-Algorithm (LBRA) that uses turn restrictions derived from the integer lattice. LBRA model uses an approach where turn prohibitions are the organization in contrast to distinct full-rank integer lattices.

Apart from the turn model, Tang et al. [43] have proposed a pressure-model-based adaptive routing algorithm. It is based on the divide-and-conquer strategy that focuses on solving congestion problems. Here the network is initially divided into regions to find routing pressure, and later regional pressure routing algorithms have selected according to pressure to depress the whole network pressure. Network pressure is also proposed as a measure of NoC performance in another work Sub-optimal Routing Algorithm (SoRA) [55] of same authors of [43]. SoRA uses the divide-and-conquer technique with network pressure to get near adaptive routing performance. Charif et al. [56] present Mini-ESPADA based on their previous work ESPADA, where packets are routed with ESPADA property in a minimal available path. Cardona et al. [58] have highlighted the need for weighted NoCs and proposed EOMesh (Even Odd Mesh), where they add a weighted allocation to heterogeneous routing that results in the nearby optimal solution. Unlike the traditional uniform interconnection between routers, Wang et al. [63] proposes the load-balanced-link-distribution strategy. It uses the source-destination node’s location to measure load factor distribution for the network.
TABLE 4. Workload distributing routing.

| Routing Algorithm       | Based On             | Using Buffer | Deadlock Freedom | Adaptivity | Path Diversity | Ref  | Year |
|-------------------------|----------------------|--------------|------------------|------------|----------------|------|------|
| Odd-Even                | Turn Model           | Yes          | Turn Restriction | Oblivious  | Minimal        | [44] | 2000 |
| Adaptive Routing        | Pressure Model       | Yes          | Turn Restriction | Adaptive   | Minimal        | [45] | 2015 |
| CHARm                   | Turn Model           | Yes          | Virtual Channel  | Adaptive   | Both           | [46] | 2015 |
| Nue                     | Dependency Graph     | Yes          | Turn Restriction | Oblivious  | Minimal        | [47] | 2016 |
| DPLR and DPLPR          | DyAD                 | Yes          | Turn Restriction | Adaptive   | Minimal        | [48] | 2016 |
| HDR                     | Deflection Routing   | No           | Deflection       | Oblivious  | Non-Minimal    | [49] | 2016 |
| PW-Odd-Even             | Odd-Even             | Yes          | Turn Restriction | Oblivious  | Minimal        | [50] | 2016 |
| MWPF                    | Whole Packet         | Yes          | Virtual Channel  | Fully Adaptive | Minimal   | [51] | 2016 |
| PEPAR                   | -                    | Yes          | Turn Restriction | Partially Adaptive | Minimal | [52] | 2016 |
| TA                      | -                    | Yes          | -                | Adaptive   | Minimal        | [53] | 2016 |
| RTM                     | Turn Model           | Yes          | Turn Restriction | Partially Adaptive | Minimal | [54] | 2017 |
| Lbra                    | Turn Model           | Yes          | Turn Restriction | Partially Adaptive | Minimal | [55] | 2017 |
| LAXY                    | Turn Model           | Yes          | Turn Restriction / eviction time | Oblivious  | Minimal        | [56] | 2017 |
| SoRA                    | Network Pressure     | Yes          | Turn Restriction | Oblivious  | Minimal        | [57] | 2017 |
| Mini-ESPADA             | ESPADA               | Yes          | Virtual Channel  | Fully Adaptive | Minimal   | [58] | 2017 |
| ZigZag                  | XY                   | Yes          | Turn Restriction | Deterministic | Minimal  | [59] | 2017 |
| EOnesh                  | XY                   | Yes          | Virtual Channel  | Adaptive   | Minimal        | [60] | 2018 |
| Modified XY             | XY                   | Yes          | Virtual Channel  | Adaptive   | Minimal        | [61] | 2018 |
| Encircle                | XY                   | Yes          | Turn Restriction | Deterministic | Minimal  | [62] | 2018 |
| WAR                     | Workload Signature   | Yes          | Virtual Channel  | Adaptive   | Minimal        | [63] | 2018 |
| Two Way Odd-Even        | Odd-Even             | Yes          | Turn Restriction | Partially Adaptive | Non Minimal | [64] | 2019 |
| Load-Balanced-Link-Distribution | Link Distribution | No          | Virtual Channel  | Oblivious  | Minimal        | [65] | 2019 |

Basic deterministic routing algorithm XY forwards packet first along x-axis & then along the y-axis, and YX does just opposite to XY. On this basis, Valencia et al. [57] have proposed ZigZag routing algorithm. In ZigZag, packets are forwarded along the axis having greater distance than other axis, and when the difference along both axes becomes same, it starts forwarding in the ZigZag way. Some modifications to XY algorithms are also present by Atik et al. [59] and Umapathy et al. [60]. In [59], authors implement buffer allocation on-demand basis, and [60] has proposed Encircle routing where the network is partitioned into regions, and then packets are forwarded in such a way that they do not enter another region. Pano et al. [61] has proposed Workload-Aware-Routing (WAR), which tries to reduce hardware overhead and increase NoC lifetime. WAR uses signatures generated from ASIC design to create a table containing port priority. This port priority helps in identifying less utilized ports to expand NoC lifetime. Some research work also considers core lifetime. In [46], authors have presented “Dynamic-Programming-based-Lifetime-aware-Routing” (DPLR) and “Dynamic-Programming-based-Lifetime-and-Performance-aware-Routing” (DPLPR) that focus on improving NoC lifetime using a budget (based on lifetime) in route decision. DPLR uses only a lifetime budget (over time resource consumption), and DPLPR uses both lifetime budget and performance as an evaluation measure. Reshma Raj et al. [62] present adaptive two-way routing in their work where they use counter to identify hot-spot cores and thereby restrict several routes to handle current network hot-spots and prevent the creation of new hot-spot. A congestion propagation network imposes extra overhead and is also complex to implement. Considering it Rohbani et al. [54] present Location-based-Aging-resilient-Xy-Yx (LAXY) that distributes workload evenly to increase NoC aging. LAXY divides the network into two halves, where it uses XY routing in one half and YX routing in the other half, which decreases traffic from the central region.

Bufferless routing is introduced as an alternative to buffered routing that helps to reduce buffer cost and its complicated allocation. Such an approach is proposed by Oxman and Weiss [47] in their work Hierarchical-Deflection-Routing (HDR). HDR is designed for hierarchical mesh where the routing algorithm forwards packets without a buffer at routers. Authors have claimed that it is the first deflection routing implementation for hierarchical mesh and present interleaving and shifting techniques. Firstly interleaving is done on a hierarchical-mesh divided into levels (like 16 × 16 for level-1, 8 × 8 for level-2, and 4 × 4 for level-3), then shifting is done by grid shift.

B. CONGESTION CONTROL

Another possible solution to address the congestion problem is to identify congestion in the network and divert traffic to the less congested area. This solution helps improve NoC performance by decreasing the transmission delay and the
TABLE 5. Congestion aware routing that use congestion propagation network.

| Routing Algorithm         | Based On        | Congestion Status | Deadlock Freedom | Adaptivity | Path Diversity | Learning | Ref | Year |
|---------------------------|-----------------|-------------------|------------------|------------|----------------|----------|-----|------|
| DyXY                      | XY              | Dedicated wire    | Virtual Channel  | Adaptive   | Minimal        | No       | 66  | 2006 |
| NOP                       | XY              | Dedicated wire    | Turn Restriction | Adaptive   | Minimal        | No       | 18  | 2006 |
| RCA                       | -               | Side Network      | Virtual Channel  | Adaptive   | Minimal        | No       | 67  | 2008 |
| DBAR                      | RCA 1D          | Side Network      | Virtual Channel  | Adaptive   | Minimal        | No       | 68  | 2011 |
| CATRA                     | RCA Quadrant    | Side Network      | -                | Adaptive   | Minimal        | No       | 69  | 2012 |
| ACO-Phd                   | ACO             | Side Network      | -                | Adaptive   | -              | No       | 70  | 2015 |
| ACO-CAR                   | ACO             | Side Network and Table Based | Virtual Channel | Adaptive   | Minimal        | No       | 71  | 2015 |
| URCA                      | DBAR            | Side Network      | Virtual Channel  | Adaptive   | Minimal        | No       | 73  | 2015 |
| ALA                       | look Ahead      | Side Network      | -                | Adaptive   | Minimal        | No       | 75  | 2015 |
| Adaptive RCA              | RCA             | Side Network      | Virtual Channel  | Adaptive   | Minimal        | No       | 76  | 2015 |
| QCA-DTC                   | DBAR            | Side Network      | Virtual Channel  | Adaptive   | Minimal        | No       | 79  | 2016 |
| FLR                       | Fuzzy Logic     | Side Network and Table Based | Virtual Channel | Fully Adaptive | Minimal        | No       | 81  | 2016 |

Congestion-Aware-and-Adaptive Algorithms

| Routing Algorithm         | Based On        | Congestion Status | Deadlock Freedom | Adaptivity | Path Diversity | Learning | Ref | Year |
|---------------------------|-----------------|-------------------|------------------|------------|----------------|----------|-----|------|
| DyXYXY                    | RCA and DBAR    | Side Network      | Virtual Channel  | Fully Adaptive | Minimal        | No       | 83  | 2017 |
| FACARS                    | RCA and DBAR    | Side Network      | Virtual Channel  | Fully Adaptive | Minimal        | No       | 89  | 2018 |
| Footprint                 | Head of Line Blocking | Side Network | Virtual Channel  | Adaptive   | Minimal        | No       | 86  | 2017 |
| Low-Latency Routing       | Traffic Monitoring | Side Network | Virtual Channel  | Adaptive   | Minimal        | No       | 90  | 2018 |
| ACDC                      | Dropping packet | Side Network      | Virtual Channel  | -          | No             | 91       | 2019 |
| MicRA                     | Side Network    | Virtual Channel   | -                | Adaptive   | Minimal        | No       | 93  | 2019 |
| Hotspot-Pattern-Aware     | West-first and North-last | Side Network | Turn Restriction | Adaptive   | Minimal        | No       | 95  | 2019 |
| CACBR                     | Cluster Based   | Side Network      | Turn Restriction & Virtual Channel | Fully Adaptive | Minimal        | No       | 97  | 2019 |
| HARE                      | Footprint       | Side Network      | Virtual Channel  | Adaptive   | Minimal        | No       | 98  | 2019 |
| SBTR and e-SBTR           | Back Pressure   | Side Network      | Turn Restriction | Adaptive   | Minimal        | No       | 107 | 2019 |
| LTCA                      | Wireless Routing | -                 | Turn Restriction | Adaptive   | Minimal        | No       | 108 | 2019 |
| Exploring SNN             | SNN             | Side Network      | Virtual Channel  | Adaptive   | Minimal        | Yes      | 109 | 2020 |
| SRM                       | SNN1            | Side Network      | Virtual Channel  | Adaptive   | Minimal        | Yes      | 110 | 2020 |
| Novel Congestion Aware Routing | Packet History | Side Network | Virtual Channel | Deterministic and Adaptive | Minimal | No | 111 | 2020 |
| Congestion Aware Routing  | CATRA           | Side Network      | Virtual Channel  | Adaptive   | Minimal        | No       | 112 | 2021 |
| Heterogeneous Congestion Criter | Novel Congestion Aware Routing | Side Network | Virtual Channel | Adaptive   | Minimal        | No       | 113 | 2021 |
| ScrN                      | RCA and NOP     | Side Network      | Virtual Channel  | Adaptive   | Minimal        | No       | 114 | 2022 |

Note: All of the above research work are based on buffered routing.

power consumption. Some techniques consider local traffic information to disburden congestion load, which means the congestion information of adjacent neighbors is used for the routing decision. However, these techniques only utilize congestion status of small sub-network. Thus, for non-uniform or bursty traffic, such an approach does not distribute traffic load uniformly across the network. Globally adaptive-routing algorithms diminish this concern by considering congestion information of larger sub-network. However, the congestion information of distant nodes gets outdated before they reach the current router. Many approaches have been proposed to manage and propagate such congestion-related information in NoC, which can be broadly classified as congestion propagation network, table entries, and flit-free header bits.

1) CONGESTION PROPAGATION NETWORK—USING DEDICATED WIRE

Table 5 depicts all algorithms that use congestion propagation network to collect congestion information. To the best of our knowledge, the first work that uses a dedicated wire to propagate congestion information is proposed by Li et al. [64] as DyXY, which uses neighborhood stress values as congestion metrics. Patti et al. [18] have further improved work proposed in [64] by adding congestion information of additional neighbors on the path toward a destination in the NoP selection strategy. This additional information provides more precise network information and improves path selection in various cases. This selection strategy utilizes neighborhood congestion status with a combination of
Odd-Even [42] routing algorithm. Gratz et al. [65] have proposed RCA that takes it further to regional congestion; unlike previous works, it propagates congestion information along an axis throughout NoC. It improves the collection of network status information which has positive impact on the NoC performance. NoP [18] relies only on local neighborhood status whereas RCA considers local neighborhood and excess of regional information. Therefore, Ma et al. [66] figure out these shortcomings and have proposed a new solution DBAR. DBAR maintains a separate register for each of dimension’s congestion status and filters them based on packet destination by considering congestion information from current to the destination node. Jose et al. [74] also figure out the drawbacks of NoP and RCA. They have combined the positive sides of both studies and have presented a hybrid adaptive model of RCA that maintains two counters for neighborhood and regional traffic, which helps in switching between selection strategies. Based on RCA and NOP, Trik et al. [112] present ScRN routing algorithm. It uses NOP when traffic is in the neighborhood and RCA when traffic is regional.

Gathering congestion information at router can be helpful in improving the performance. However, on the other hand, it also results in hardware overhead. Ebrahimi et al. [67] have presented CATRA that gathers more congestion information using less wire. To collect this congestion information, they create an agent node for a group of four nodes and connect those agent nodes to exchange congestion information. CATRA also assigns proper weights to congestion data of a node according to the distance from the current node. Zong et al. [71] present an unbiased RCA that is an improvement of DBAR. It covers more nodes to collect congestion status than DBAR algorithm. RCA and DBAR assign equal weights to both nearby and distanced nodes. Touati and Boutekkouk [81] have proposed DyXYYX routing algorithm. DyXYYX takes congestion information from both ends and assigns them a halved weight at each hop from source to destination and destination to source. Another work FACARS, has been proposed by Touati and Boutekkouk [87], solves it by utilizing neighborhood and global information simultaneously. The FACARS presents two versions representing node congestion status in two and three-level.

Congestion information utilized in previous research is of two types, spatial and temporal. Most researchers use only one of them as network congestion status. Hsin et al. [68] present a Network-Information-Region (NIR) framework that integrates both spatial and temporal information and proposes ACO-PhD routing. NIR propagate spatial information as \( PH_{acc} \) and temporal information as \( PH_{Dtg} \) which are controlled with parameters \( \alpha \) and \( \beta \). Simulation results of ACO-PhD show that the RCA and other algorithms performance can be achieved by adjusting these parameters. However, ACO-PhD routing consumes excessive hardware, which results in increased cost. Chang et al. [69] suggest a solution as ACO-CAR. It improves ACO by removing excess information and merging table entries in the group. To reduce the computation required for selecting output direction, Menon et al. [73] have proposed Adaptive-Look-Ahead (ALA) algorithm. ALA makes routing decisions at the alternative node using the lookahead approach, which decreases the computational selection requirement. Another work proposed by Xu et al. [77] highlight this overhead consumption in collecting quantitative information and presents the QCA technique. QCA_DTC uses a single wire to collect quantitative information for each node by sending the variation in congestion status rather than complete status. Tutas and Chrysostomou [79] present Fuzzy-Logic-Routing (FLR) that is a fuzzy-based approach that considers only neighborhood traffic load to make route decision. The traffic pattern is not constant as they vary with time, affecting the routing algorithm’s performance. Also, various routing algorithms do well for specific traffic; thus, getting better performance with one routing algorithm for all traffic patterns is not always possible. Ul Mustafa et al. [113] present an Adaptive-Routing-Framework that makes use of XY, negative-first, and west-first. It initially identifies traffic patterns and then selects the routing algorithm accordingly.

Dividing big problems into more minor ones can help as the smaller problem is easy to solve and then can be merged with other smaller solutions to get a solution for the big problem. Akbar and Safaei [80] have proposed Congestion-Aware-and-Adaptive-Routing algorithm that divides the network into smaller sub-network (3-4 nodes per dimension). Firstly, it selects a sub-network according to the destination, then picks a node from its boundary as a temp node to forward the packet, and when finally the packet reaches the target sub-network, it delivers it to the destination node. Another research paper presented by Taherkhani et al. [110] also follow the same strategy to segment the network into sub-network of equal size and then uses global and local routing to improve latency and throughput. It moves packets from one sub-network to another by selecting a boundary node, and when the packet reaches its destination sub-network then moves the packet to its destination node. Fu and Kim [84] have highlighted the effect of virtual channel occupancy as a thick branch in the congestion tree and proposed Footprint-Routing-Algorithm to overcome it. Footprint routing minimizes congestion thickness by selecting the output port with the highest free VC in normal traffic flow. However, in congested traffic, a packet will start forwarding to the port allocated to the packet of the same destination. Footprint prevents the congestion tree from thickening to some extent while considering only the current path is not enough, thus exploring path history.

Jin et al. [96] have proposed History-Aware Adaptive Routing Algorithm for Endpoint Congestion (HARE). HARE solves end-point congestion using path history by calculating the depth footprint for VC. Another research work on the history of the packet is presented by Akbar and Safaei [109]. It uses the packet’s history & adaptivity and stores this information in the packet header. It generally works deterministically; however, it adaptively changes the path to reduce congestion when congestion appears. A heterogeneous
TABLE 6. Congestion aware routing (Table Based).

| Routing Algorithm               | Based On            | Congestion Status | Deadlock Freedom | Adaptivity       | Path Diversity | Learning | Ref   | Year |
|---------------------------------|---------------------|-------------------|------------------|------------------|----------------|----------|-------|------|
| QCA                             | Q-learning          | Table Based       | Virtual Channel  | Adaptive         | Minimal        | Yes      | [92]  | 2011 |
| CrQ and PCrQ                    | Q-learning          | Table Based       | Virtual Channel  | Adaptive         | Non-Minimal    | Yes      | [72]  | 2015 |
| Cognitive based Routing         | Q-learning          | Table Based       | Virtual Channel  | Adaptive         | Non-Minimal    | Yes      | [80]  | 2016 |
| AQFE                            | Full Echo Q-Routing | Table Based       | Virtual Channel  | Adaptive         | Minimal        | Yes      | [84]  | 2016 |
| AQKERM                          | AQFE                | Table Based       | Virtual Channel  | Adaptive         | Minimal and    | Yes      | [85]  | 2017 |
| PreNoC                          | ANN                 | Table Based       | Virtual Channel  | Oblivious and    | Adaptive       | Non-Minimal | [87] | 2017 |
| HTOE                            | Q-learning and Odd-Even | Table Based | Turn Restriction | Partially Adaptive | Partially Adaptive | Minimal | Yes  | [88] | 2017 |
| Adaptive Q-routing              | Q-learning          | Table Based       | Virtual Channel  | Partially Adaptive | Partially Adaptive | Minimal | Yes  | [94] | 2019 |
| Switching at Flit Level         | Flow control        |                   | Virtual Channel  | -                | Minimal        | No       | [116] | 2020 |
| Reinforcement Learning Enabled Routing | Reinforcement Learning | Table Based | Virtual Channel  | Adaptive         | Minimal        | Yes      | [117] | 2021 |
| HiFMP                           | History Based Ant Colony Algorithm | Table Based | Virtual Channel  | Adaptive         | Minimal        | No       | [118] | 2021 |
| DeepNR                          | Reinforcement Learning | Table Based | Virtual Channel  | Adaptive         | Minimal        | Yes      | [119] | 2022 |

Note: All of the above research work are based on buffered routing.

congestion criterion, presented by Akbar and Safaei [111] maintains a separate congestion threshold for different nodes of NoC. This criterion is assigned offline based on the graph theory criterion (Betweenness Centrality) for traffic passing through it.

Li et al. [88] monitor traffic for congestion status along with the amount of traffic propagated. The latter helps in predicting the latency at each router. Both load and latency can control traffic flow. Xiao et al. [89] have proposed a strategy of aggressively taking down the congestion and latency in NoC. They propose a lightweight control mechanism based on flow prediction, consisting of global and local control. The study selectively drops and recovers data in the qualitative model. A back pressure-based work is presented by Deb et al. [105] that focuses on reducing hop count by utilizing a long-distance transmission line. They propose SBTR and e-SBTR router architectures. Gaffour et al. [91] propose a Minimal-Congestion-aware-Routing-Algorithm that utilizes local congestion information for structuring global routing strategy in 2D/3D NoC. It computes free buffer status and stores it in the flag as three levels: highly congested, congested, and non-congested. Communication for long-distance nodes is studied by Mamaghani and Jamali [106], that proposes LTCA. It utilizes wireless routing based on the time interval for allocating a wireless channel to long-distance nodes.

For synthetic traffic, hot-spot pattern is closer to real traffic scenario. Luo et al. [93] present Hotspot-Pattern-Aware-Routing Algorithm that contains a dedicated mechanism for detecting a hot-spot pattern. It uses a hot-spot block at each router to detect hot-spot traffic. Cluster-based routing is proposed by Bahman et al. [95] in their work Congestion-Aware-Cluster-Buffer-base-routing-Algorithm (CACBR). It divides network in the cluster and calculate free-buffer slot for each direction, then output direction is selected according to free-buffer slots.

Developing a machine learning model for congestion prediction is also very helpful, although still a rarely explored field. Javed et al. [107], [108] explore spiking neural network (SNN) for congestion prediction in NoC that helps in learning and identifying the temporal nature of traffic. They propose a router and network-level congestion prediction model based on Spike Response Model. They claim to predict congestion in the network 30 cycles before its appearance.

2) TABLE BASED—USING TABLE ENTRIES

Apart from propagating congestion information, some research works use tables at each router to maintain congestion information of routers. Table 6 shows all algorithms that store congestion information in congestion tables.

Q-learning is a reinforcement learning technique that uses table entries to select an action from the current state. Such strategies have rarely been utilized in NoC, so Farahnakian et al. [90] propose a Q-learning-based-Congestion-aware Routing algorithm that utilizes network congestion information with Q-learning to provide a less congested path. QCA uses reinforcement learning for updating table entries in the form of Q (y,d) (Q-value). The Q-value represents latency to reach destination d with router y, and the algorithm has to select router y with the lowest Q-value (or latency). An improvement of QCA as Credence-based-Q-routing & Probabilistic-Credence-based-Q-routing have been proposed.
by Gupta et al. [70]. These use credence as a c-value to measure Q-value and allow the learning rate to vary with the time that improves the learning process.

Gupta and Bhargava [78] introduce cognitive in the network, which helps to find environment changes in NoC; for this, the method monitors congestion in the network to collect information that helps in decision making. Another work that changes learning rate is presented by Shilova et al. [82] as adaptive-Q-routing-technique-with-Full-Echo-extension. AQFE uses two rates for the learning process and switches between them based on average delivery time. It solves irregular packet latency in Full-Echo routing. Improvement during settling and overshooting time in the learning process of AQFE is presented by Kavalerev et al. [83] as Adaptive-Q-routing-with-Random-Echo-and-Route-Memory (AQRERM). AQRERM presents two advances in Q-learning; first, it selects a set of random neighboring nodes to update the Q-value, and second, it allows the packet to record the neighborhood status of visiting node. Krishman et al. [86] present Hybrid-Odd-Even-Q-Routing (HOEQ) that maintains the Q-table to store latency in the form of Q-values and forwards packet accordingly. It utilizes Odd-Even restriction to avoid deadlock. Another Q-learning-based routing algorithm is proposed named as dynamical-Q-routing by Fan et al. [92]; it uses real-time traffic monitoring with less historical data to forward the packet.

The reinforcement learning-based routing strategy is presented by Reza and Le [115], which maintains states and, based on those states, action is performed. In the action part, it selects a routing algorithm (i.e., XY-routing, adaptive west-first routing, random-oblivious routing) based on the network congestion status. RS et al. [117] present DeepNR and shows the effectiveness of reinforcement learning in designing the policies for NoC. It represents network congestion information as state, output directions as action, and delay in a queue as a reward. Apart from using Q-learning in NoC, Kinsy et al. [85] use a neural network and have proposed PreNoC (Predictive-Routing-for-NoC) routing algorithm. PreNoC uses a different approach from other learning methods; here, the learning process is divided into three parts which iterate when required. In step-1, congestion information is collected. In step-2, it observes and validates that information. Last step, entries of tables are updated accordingly. It works in oblivious mode and checks network performance at a regular time. When performance degradation is observed, it performs the learning phase to improve. Alaei and Yazdanpanah [116] present parameterizable router HiFMP and uses deterministic XY and history-based routing. Based on congestion status, it checks the table for last path usability and, if needed, finds a more efficient path and updates the table accordingly.

3) FLIT HEADER—USING FREE HEADER BITS
In the NoC flit header, more than 52 free bits are available, which may be used for propagating congestion information. It does not consume additional resources, thus providing an economical way to propagate information. Table 7 depicts all algorithms that use header bits to propagate congestion information.

Header bits are utilized by Liu et al. [72] in their work FreeRider. It uses such free header bits to propagate the rich congestion status of routers and nodes up to the destination. It propagates congestion information in both axes, which are further improved by Ramakrishna et al. [75] in their proposed algorithm Global-Congestion-Aware. GCA uses piggy-backing to timely propagate congestion status throughout the NoC, providing a complete network congestion status. In GCA, when a flit reaches a router, congestion information of other routers is extracted from it, and its congestion information is appended to the flit header. GCA is further enhanced by Yan [76] in their proposed work Enhanced-Global-Congestion-Aware. EGCA improves in propagating congestion information by updating neighboring nodes rather than only visited nodes, thus decreasing the staleness of congestion information and improving performance. GCA-based another work is presented by Fang et al. [118] as partition-based-congestion-aware-routing (PAR); they divide the network into central and edge regions with low and high priority, respectively. PAR utilizes flit-free bits to propagate congestion information. Considering extra hardware cost added by the virtual channel that increases with network size, Han et al. [94] present the solution as Low-Cost-Congestion-Detection-Mechanism based on the turn model (LCCDM). It uses buffer occupied value as congestion metrics which is propagated using flit header free bits. Ahmad et al. [119] present a congestion-aware routing algorithm that uses data flit to propagate congestion information and uses that information in distributing traffic to avoid congestion situation.

C. WORKLOAD DISTRIBUTION AND CONGESTION CONTROL
Both workload distribution and congestion control are practical solutions that address congestion problems separately. However, their combination can further resolve congestion issues. Congestion control implementation requires additional time for calculation and execution that adds up extra latency, so adopting such a control mechanism when needed is a good strategy [97]. Some proposed works have adopted (as shown in Table 8) both the strategies, i.e., distributing workload in regular traffic and switching to congestion control when growing traffic creates congestion in NoC.

Hu and Marculescu [97] show that for a non-congested network, the performance of XY is much better than Odd-Even. However, in congestion situations, Odd-Even outperforms XY. Influencing XY’s performance in regular traffic, they have proposed DyAD, which strives to blend the positive side of both deterministic and adaptive algorithms by switching between them with changing traffic flow. DyAD is activated after congestion occurs, but Lotfi-Kamran et al. [98] adopt the other way by avoiding congestion before it appears. In their work BARP, they distribute the load evenly in a
congestion-free situation by dividing traffic in each direction whenever possible. However, when congestion starts to appear, it generates an additional routing packet to avoid it.

In network-centric congestion and self-similar traffic, latency usually increases, which must consider improving performance. To address this, Ni and Liu [100] present Distance-Prediction-XY routing (DPXY). It initially uses the routing function to find the available direction to forward the packet. The selection function is used to select one direction based on buffer status. Ramanujam [99] has presented Destination-Based-Adaptive-Routing that uses strategy to store destination delay information of every node at each router and uses this information to forward packets. Increased aging of some routers creates age difference between cores, Alshraiedeh and Kodi [101] present an Aging-Aware-Routing algorithm that distributes load to the less utilized node for the increased lifetime of NoC with negligibly affecting performance. For this, they use directional age measure as packet-per-port ($P^3$) that is used along with congestion score (similar to RCA-1D) for making a routing decision.

### D. CONGESTION CONTROL AND FAULT TOLERANCE

A single fault in the network can degrade the performance of NoC, which sometimes even leads to system failure. Fault tolerance routing is the natural solution for a faulty network to facilitate reliable on-chip communication. Many researchers have targeted the problem of congestion control and fault tolerance separately. However, some works (as shown in Table 9) consider fault tolerance and congestion control because both divert the traffic on alternative routes that may significantly affect NoC performance.

Gupta et al. [102] focus on both congestion and fault and propose $\sigma$LBDR as an improvement of uLBDR to handle the varying level of congestion in NoC. $\sigma$LBDR can address congestion and fault by using 16 $\sigma$-bits for congestion of two-hop nodes to minimum information in the routing table. Instead of using a table, TRACK [121] is based on $d^2$-LBDR (logic-based routing) that can tolerate single and multi-link faults. It re-configures only fault-affected nodes, whereas other unaffected nodes can continue to work as usual. Gabis et al. [103] have proposed Heuristic-based-Routing-Algorithm (HRA) based on the A-Star search algorithm. It uses neighbor router local information (congestion rate and latency) and finds a path by constructing a tree with the initial state as the source node and the target state as the destination node. Shafiei and Sattari-Naeini [120] present a congestion-aware and fault tolerance routing algorithm that collects neighboring congestion and fault information, and based on that, an appropriate output direction is selected. Based on process variation, Muhammad et al. [104] have proposed Congestion-aware-fault-tolerant-and-process-variation-aware routing that maintains two tables: first to store delay-based directions and second to store delay information in the queue. The direction obtained helps reduce traffic, and delay information at the router is utilized to avoid congestion. An NoC framework based on deep reinforcement learning (DRL) is presented by Wang and Louri [122]; CURE can tolerate transient and permanent faults and improves latency and energy efficiency. It uses DRL to update the dynamic control policy at each router based on NoC behavior and accordingly select operation mode. Kumar et al. [123] have proposed Fault tolerant and Congestion Aware Routing (FTCAR). It uses Duato’s theory to prove deadlock freedom.
TABLE 9. Congestion control and fault tolerance routing.

| Routing Algorithm | Based On       | Congestion Status | Deadlock Freedom       | Adaptivity      | Path Diversity | Learning | Ref   | Year |
|-------------------|----------------|-------------------|------------------------|----------------|----------------|----------|-------|------|
| ωLBDR             | LBDR           | Table Based       | Virtual Channel        | Adaptive       | Minimal        | No       | [104] | 2015 |
| HRA               | A-Star Search  | Side Network      | Heuristic Tree         | Fully Adaptive | Minimal        | No       | [105] | 2016 |
| Adaptive Multipath Routing | DyAD | Side Network | - | Adaptive | Minimal | No | [122] | 2018 |
| CFPA              | Process Variation | Side Network | Hop count Priority | Adaptive | Minimal | No | [106] | 2019 |
| TRACK             | $d^2$-LBDR     | Logic Based       | Routing Restriction    | Adaptive       | Minimal        | No       | [123] | 2020 |
| CURE              | deep reinforcement learning | Side Network | Virtual Channel | Adaptive | - | Yes | [124] | 2020 |

Note: All of the above research work are buffered based routing and are not using machine learning except CURE.

using acyclic Channel Dependency Graph. It can tolerate all single link faults.

IV. IMPLEMENTATION DETAILS AND SELECTION OF STATE-OF-ART FOR EXPERIMENTATION

The best solution to avoid congestion is the proper and even distribution of traffic. Research works that distribute traffic without considering congestion information are listed in Table 4. Odd-Even [42] is a state-of-art solution that uses turn restriction to distribute traffic evenly. However, ideal traffic distribution is not possible without exact and timely network traffic or congestion information. It makes researchers focus on providing the network’s actual congestion status to divert traffic accordingly. Thus, most of researchers are focusing on collect of congestion information using dedicated wire as shown in Table 5. However, processing of this congestion information to provide a correct view of the network is also crucial. Thus, assigning the right weight to various congestion information (i.e., local and regional) is still an important area of research, along with the proper and timely distribution of congestion information.

Implementation of the congestion collection network puts extra overhead on NoC. Considering this additional overhead, some researchers try to minimize the size of congestion information during transmission to reduce such costs. However, others use free packet bits to forward that information without additional wiring. Such research works are listed in Table 7. Apart from dedicated side networks and free header bits, some researchers adopt machine-learning approaches and use tables to store output port priority information, specially Q-learning, as shown in Table 6. Some additional latency is introduced during the calculation and processing of congestion information. Thus, it is better to use it when needed [97]. It shows the importance of proper workload distribution initially without processing congestion information that helps to avoid congestion situations, and process congestion information only when congestion starts appearing. Such research work is shown in Table 8 that uses both strategies of workload distribution and congestion control, and event work switches between them intelligently to save resources.

A. SIMULATORS

Various simulators are used to simulate and analyze routing and selection strategies. We summarize all simulators in Fig. 4 according to their uses in the refereed research papers. Most research works preferred Noxim-simulator, developed in C++ using systemC library; Booksim-simulator is the next most popular simulator; next to them are Nigam and Gem5 simulators that are noticeably used by research works, some other work uses in-house or other simulators.

B. EVALUATION METRICS

Evaluation metrics are the tools to evaluate and measure the performance of research works. Using these evaluation metrics to verify routing and selection algorithm performance is crucial for NoC optimization. A proposed routing or selection algorithm should guarantee low latency, high throughput, less power or energy consumption, and minimum area cost for designing NoC. In Fig. 5, the X-axis shows the various performance parameters, and the Y-axis indicates the number of research works in this review paper that uses these parameters. It shows that more attention is toward latency and throughput for evaluating algorithms performance compared to power, area, and other metrics. However, power and area costs in chip manufacturing also need to be reduced, as they are critical for NoC designing.
NoC performance varies with traffic scenarios. These various traffic loads can be classified into two broad categories: synthetic and real traffic. Synthetic traffic is system-generated traffic that follows a fixed pattern, whereas real traffic is traces collected from the execution of a real-world application that mimics real traffic. The use of this synthetic and real traffic for NoC performance evaluation is shown in Fig. 6, where the Y-axis indicates the number of research works in this review paper that uses these traffic patterns. Here, blue and red colors are used for synthetic and real traffic, respectively. Uniform random and transpose are the most used traffic patterns and hot spot, bit reversal, and shuffle traffic are less utilized under synthetic traffic. From Fig. 6, we can see that synthetic traffic has been used far more than real traffic to evaluate NoC performance in most research papers. However, to validate the NoC performance under the actual working condition, it should be tested in real traffic scenarios. SPLASH-2 and multimedia-system/video are the commonly used traffic patterns for real traffic.

V. EXPERIMENTAL ANALYSIS OF STATE-OF-ART

Initial NoC research focuses only on routing strategies. However, as the number of cores and workload increases in NoC, the congestion problem comes into the picture. Succeeding work starts focusing on selection strategies to address this issue by reducing such congestion situations. For this, it needs to collect the congestion status of the network accurately and efficiently. Initial works consider congestion status of only connected neighbor as shown in Fig. 7a. NOP [18] includes congestion condition of neighbors on the path as shown in Fig. 7b which significantly improves network performance. RCA [65] added regional congestion status in the selection of direction as shown in Fig. 7c. This regional information with local information is beneficial for output direction selection. However, in regional congestion data, it also considers the congestion status of routers that lies outside of the source-destination path. DBAR [66] improves it by removing the router’s congestion status that lies outside the source-destination path using destination information as shown in Fig. 7d. However, it only considers along with axis congestion status of routers. CATRA [67] further improves it by taking all possible minimum path congestion statuses as shown in Fig. 7e. Separate from them, CACBR [95] visualizes congestion status in the form of a cluster as shown in Fig. 7f. It utilizes cluster congestion status for the selection of output direction.

To evaluate the selection capability of various strategies for congestion information collection, we implement NOP, DBAR, CATRA, and CACBR. To analyze machine learning for NoC performance improvement, we implement Q-learning on congestion information collected by DBAR. An event-driven cyclic accurate Noxim [124] simulator is modified to model and perform simulation. Average latency and throughput are used as a metric for algorithm performance analysis. The term latency is the measure of time consumed by a packet from entering the network (from the processing element at the source node) till it leaves the network (to the processing element at the destination node). The term throughput is the measure of the rate to deliver packets per node per unit of time. Simulation environment details have been provided in table 10.
A. SYNTHETIC TRAFFIC

Synthetically generated traffic facilitates the simulation of NoC. Such synthetic traffic helps NoC performance analysis by stressing the network in several ways. Under these traffic patterns, we can put pressure on a specific network area to find NoC bottlenecks at various injection rates for maximizing throughput and minimizing latency.

In our implemented algorithms, NOP and CACBR rely totally on neighboring local congestion information for output direction selection, whereas DBAR and CATRA try to give the right weight to both local and regional congestion. Analysis of these algorithms under various traffic patterns is presented below.

1) RANDOM TRAFFIC

The random traffic pattern is generated by sending a packet to a random node with uniform probability. The results presented in Fig. 8a and Fig. 9a shows that at the beginning of the simulation, congestion is low and local information is very effective. However, as the simulation progresses and congestion appears in most parts of the network, regional information gains significant importance, and the average latency and throughput of DBAR and CATRA show improvement over NOP and go near CACBR performance. CATRA efficiently utilizes regional information that improves its performance as congestion increases. The use of Q-learning on congestion information improves its performance over DBAR. However, it still lags over other algorithms.

2) TRANPOSE1 TRAFFIC

In transpose1 traffic, traffic pattern is generated by sending packets from node (x,y) to node (n-(y+1), n-(x+1)) where the destination and source count are higher in positive and negative traffic, respectively. The average latency result presented in Fig. 8b shows that local information is adequate for low congestion. However, when congestion increases, regional information starts playing its role. NOP and CATRA perform better initially, whereas DBAR and CATRA performance get better when congestion grows. The throughput result in Fig. 9b shows that CATRA, NOP, CACBR, and DBAR performance well in order, respectively. Q-learning show improved performance over DBAR; however, it is still underperforming compared to others. This result shows the importance of neighboring congestion information for performance improvisation.

3) TRANPOSE2 TRAFFIC

Transpose2 traffic pattern is generated by sending packets from node (x,y) to node (y,x) where the destination and source count are higher in positive and negative traffic, respectively. The average latency result presented in Fig. 8b is similar to transpose1 traffic. Here, local information is effective in low congestion, and regional information is effective in high congestion. However, in transpose2 traffic, improvement of DBAR and CATRA over NOP and CACBR is low. The throughput result in Fig. 9b shows that CATRA, NOP, and CACBR perform nearly equally and better than DBAR. Q-learning shows significant performance improvement over DBAR. However, lack of proper congestion information limits its performance.

4) SHUFFLE TRAFFIC

In shuffle traffic, pattern is generated by sending packets from source to destination, where the destination is determined by...
shifting one bit left to the source node. The results presented in Fig. 8d and Fig. 9d show that initially, the performance of NOP is better than all other algorithms. CATRA is better than DBAR and CACBR, and CACBR is better than DBAR. NOP performance has improved over others as congestion increases. However, after a certain point, when congestion further grows, NOP performance suddenly decreases over other selection strategies. Q-learning also shows noticeable performance improvement over DBAR.

5) BIT-REVERSAL TRAFFIC
Bit-reversal traffic pattern has generated by sending packets from node \((x, y)\) to node \((n-x, n-y)\) and then reversing it (i.e. sending packets from node \((n-x, n-y)\) to node \((x, y)\)). The results presented in Fig. 8e and Fig. 8e are similar to shuffle traffic. NOP and CACBR performance lines are more curved than DBAR and CATRA. It indicates that with increasing congestion conditions, the performance of NOP and CACBR decreases progressively compared to DBAR and CATRA, highlighting the importance of regional congestion information under high congestion. Q-learning shows minor performance improvement over DBAR.

6) BUTTERFLY TRAFFIC
Butterfly traffic pattern has generated by sending packets from node \((x(n-1), x(n-2)...x1, x0)\) to node \((x0, x1...x(n-2), x(n-1))\). The average latency result presented in Fig. 8f shows that initially, the performance of all algorithms is equal, whereas as congestion increases, the performance of CATRA and CACBR improves over NOP and DBAR. However, when congestion grows more, the performance of DBAR and CATRA have improved over NOP and CACBR. The throughput result in Fig. 9f shows that NOP, DBAR, and CACBR perform nearly equally. However, CATRA shows significant performance improvement over them. Butterfly traffic is the only case for synthetic traffic where Q-learning
FIGURE 9. Performance in terms of throughput under different synthetic traffic loads in 8 × 8 2D-mesh.

shows performance improvement over DBAR, NOP, and CACBR.

Simulation results highlighted the importance of local congestion information under low congestion conditions, whereas under high congestion conditions, regional information importance increases. However, as congestion grows in NoC, regional congestion information helps divert traffic toward the less congested region. NOP uses the congestion status of neighbor and neighbor-on-path for output direction selection. NOP initially shows good performance when congestion is low. However, as congestion gradually increases, network efficiency decreases significantly under random and bit-reversal traffic. CACBR uses the congestion status of the neighbor cluster. It performs well under random traffic, whereas its performance could be better not up to the limit under all other traffic patterns. DBAR uses local and regional (along the axis) congestion status. In low congestion conditions, DBAR underperforms in comparison to all other selection strategies. However, as congestion starts growing in NoC, its performance improves over NOP and CACBR. It highlighted the importance of regional information for improving performance under highly congested NoC. CATRA can be visualized as an improvement over DBAR, as it collects the congestion information of local and regional areas within the minimal path. Its performance under low congestion is better than DBAR and CACBR for all traffic patterns (except random traffic, where CACBR performance is better over others), and as traffic grows in NoC, it starts to perform much better over other selection strategies for all traffic patterns. Q-learning is one of the most used learning techniques for congestion control and performance improvement. Q-learning implementation on congestion information (DBAR congestion information for our experiment) shows performance improvement. However, the performance is not
much improved in comparison to CATRA. Q-learning performance highlights that machine learning can improve performance, whereas proper congestion information can provide far better performance.

**B. REAL TRAFFIC TRACES**

Trace-driven simulations are used as an alternative to complete real system simulations. Such system traces have been taken from real application implementation [125]. These traces help to find NoC bottlenecks and impact routing and selection strategies under the real application. For such traffic simulation, we use netrace traffic traces taken from PARSEC benchmark suits [126] M5 simulation.

1) **BLACKSCHOLES TRACE**

Blackscholes trace has been taken from Intel RMS benchmark mark that utilizes Black-Scholes partial differential equation for the price calculation in European options [126]. The results presented in Fig. 10a and Fig. 11a show that initially, NOP, DBAR, and CACBR start with the same performance. However, with simulation progress, NOP and CACBR equally improve over DBAR, which shows the importance of local congestion status. CATRA uses congestion information of all possible paths with justified probability, which helps it to outperform all other selection strategies. For real traffic, Q-learning shows the same performance improvement as synthetic traffic.

2) **BODYTRACK TRACE**

Bodytrack trace has again from Intel RMS benchmark mark that utilizes some camera’s [126]. The results presented in Fig. 10b and Fig. 11b show that all selection strategies perform equally well under low congestion situations. However, with the increase in congestion, DBAR performance decreases, whereas CACBR and NOP performance slightly improve over CATRA. Q-learning improves performance over DBAR; however, it still lags behind other algorithms performance. This improved performance of NOP and CACBR over DBAR and CATRA highlights the importance of local information.

3) **CANEAL TRACE**

Canneal trace has been taken from Princeton University Ferret developed kernel that drives fine-grained parallelism with highly fast-growing synchronous strategy [126]. The results presented in Fig. 10c and Fig. 11c show that at the beginning of the simulation, NOP, DBAR, and CACBR start with the same performance. However, as the simulation progresses, NOP and CACBR equally improve over DBAR. CATRA initially underperforms under low congestion situations. However, when congestion grows in the network, it again outperforms all other selection strategies. Q-learning helps to improve performance over DBAR. However, it underperforms compared to NOP, CATRA, and CACBR algorithms.

4) **FERRET TRACE**

Ferret trace has been taken from Princeton University Ferret toolkit that is used to find the similarity in content. Here it is configured for searching image similarities [126]. The results presented in Fig. 10d and Fig. 11d show a similar outcome to the bodytrack trace that strengthens local information importance.

5) **FLUIDANIMATE TRACE**

Fluidanimate trace has been taken from Intel RMS benchmark mark that utilizes Smoothed Particle Hydrodynamics for interactive fluid animation simulation [126]. The average latency result presented in Fig. 10e shows that DBAR underperforms from starting to end. At the beginning of the simulation, NOP performs slightly better than CATRA and CACBR. However, as the simulation progresses and congestion increases in the network, NOP performance starts decreasing. The throughput result in Fig. 11e shows that DBAR underperforms, and NOP, CATRA, and CACBR perform equally well. These results emphasize that local congestion information is essential in low congestion situations. However, nearby area congestion information starts gaining importance with increased congestion. The performance of Q-learning is better than DBAR and lower than other algorithms. However, with the increase in congestion, its performance goes near NOP, CATRA, and CACBR. It shows the machine learning potential to improve the performance.

6) **SWAPTIONS TRACE**

Swaptions trace has been taken from Intel RMS benchmark that utilizes Heath-Jarrow-Morton for price calculation of swaptions by employing Monte Carlo method [126]. The results presented in Fig. 10f and Fig. 11f are nearly similar to Fig. 10b and Fig. 10d, as DBAR performance decreases when compared to NOP, CATRA, and CACBR. It shows the staleness of long-distance nodes that need to be avoided. Q-learning performance is also similar to its performance in Bodytrack.

7) **VIPS 64C TRACE**

Vips trace has been taken from VASARI-Image-Processing-System that uses basic image processing for on-demand print service [126]. The results presented in Fig. 10g and Fig. 11g show that DBAR has performed worst in all of them. CACBR performs slightly better than CATRA and NOP, whereas, with simulation progress, CATRA starts improving over CACBR, whereas NOP outperforms all of them. Thus, in such a situation, local information helps to improve performance over regional information. Q-learning performs better than DBAR and surpasses other algorithm performance with increasing congestion. It highlights machine learning’s capability to improve results.

8) **X264 TRACE**

X264 trace has been taken from H.264/AVC (Advanced-Video-Coding) video encoder that uses a lossy compression technique [126]. The results presented in
Fig. 10h and Fig. 11h show that DBAR underperforms from start to end. NOP, CATRA, and CACBR perform equally in the first half of the simulation, whereas in the second half of the simulation, NOP and CACBR improve over CATRA. Q-learning shows improved results over DBAR. However, it still lags behind NOP, CATRA, and CACBR.

Simulation results from different real traffic traces highlighted that regional information collected along the axis...
is not of much use, as there is a high probability that they may get stale over time before reaching a long-distance router. Local information either from neighbor-on-path or local clusters shows a noticeable improvement in performance in nearly all traffic patterns. However, results are much better when local information has been used with non-local information (up to a specific node so that information will not stale) in justified weights, especially in

FIGURE 11. Performance in terms of throughput under different real traffic traces in 8 × 8 2D-mesh.
BlackScholes and channel traces. The use of machine learning further improves the performance of the network. However, it performs near NOP, CATRA, and CACBR in canneal and swaptions traffic.

VI. CONCLUSION AND FUTURE WORK

This paper summarizes the significant routing and selection algorithms implementation and analysis. These algorithms have targeted handling congested traffic situations in NoC by adopting various strategies. Workload distribution strategy smartly distributes traffic load throughout the network. However, varying traffic still creates congestion situations, so including congestion information will effectively improve traffic distribution in NoC. Such congestion status can be categorized into local and regional congestion, where each significantly affects the network performance. So designing a selection algorithm needs to provide the correct weight to each type of information for efficient output direction selection. Sharing this local and regional congestion information among routers is also a challenging issue solved by adopting a side network or using a packet header. However, both techniques have limitations as they require additional hardware overhead and timely updating of information before they become stale, which is challenging. Machine learning, mainly Q-learning, is also utilized by some researchers to predict the correct congestion status of the network, which has been implemented by maintaining a table at each router, and their table entries are used for output direction selection.

Many researchers worked on various techniques to avoid congestion before it appears in the network and handles congestion after it occurs. However, it is still possible to improve it. Since local congestion needs to be prioritized, it can effectively use side networks to disseminate local and neighboring information. It can use a packet header to reduce hardware overhead and provide bulk information on network regional congestion status. There is much scope for applying machine learning to predict the network’s regional status to solve the regional congestion information staleness problem. Simultaneously, machine learning can also learn traffic behavior and modify routing and selection strategies accordingly.

REFERENCES

[1] L. Benini and G. De Micheli, “Networks on chips: A new SoC paradigm,” Computer, vol. 35, no. 1, pp. 70–78, Jan. 2002.
[2] W. J. Dally and B. Towles, “Route packets, not wires: On-chip interconnection networks,” in Proc. 38th Design Autom. Conf., 2001, pp. 684–689.
[3] M. Daneshtalab, M. Ebrahimi, P. Liljeberg, J. Plosila, and H. Tenhunen, “A low-latency and memory-efficient on-chip network,” in Proc. 4th ACM/IEEE Int. Symp. Netw. Chip, May 2010, pp. 99–106.
[4] D. Wu, B. M. Al-Hashimi, and M. T. Schmitz, “Improving routing efficiency for network-on-chip through contention-aware input selection,” in Proc. Asia South Pacific Conf. Design Autom., 2006, pp. 1–6.
[5] A. Adriahentainena, H. Charlery, A. Greiner, L. Mortier, and C. A. Zefirino, “SPIN: A scalable, packet switched, on-chip micro-network,” in Proc. Design, Autom. Test Eur. Conf. Exhib., 2003, pp. 70–73.
[6] M. Forsell, “A scalable high-performance computing solution for networks on chips,” IEEE Micro, vol. 22, no. 5, pp. 46–55, Sep. 2002.
[7] M. Mosadeli, P. P. Maji, and W. Vanderbauwhede, “Design and implementation of the Quarc network on-chip,” in Proc. IEEE Int. Symp. Parallel Distrib. Process., May 2009, pp. 1–9.
[8] J. Balfour and W. J. Dally, “Design tradeoffs for tiled CMP on-chip networks,” in Proc. ACM Int. Conf. Supercomputing 25th Anniversary Volume, 2006, pp. 390–401.
[9] I. Kim, J. Balfour, and W. Dally, “Flattened butterfly topology for on-chip networks,” in Proc. 40th Annua. IEEE/ACM Int. Symp. Microarchitecture (MICRO), Dec. 2007, pp. 172–182.
[10] M. B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, J.-W. Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shidman, V. Strumpen, M. Frank, S. Amarasinghe, and A. Agarwal, “The raw microprocessor: A computational fabric for software circuits and general-purpose programs,” IEEE Micro, vol. 22, no. 2, pp. 25–35, May 2002.
[11] P. Gratza, C. Kim, R. McDonald, S. W. Keckler, and D. Burger, “Implementation and evaluation of on-chip network architectures,” in Proc. Int. Conf. Comput. Design, Oct. 2006, pp. 477–484.
[12] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, and N. Borkar, “An 80-tile 1.28 TFLOPS network-on-chip in 65 nm CMOS,” in Proc. IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers, Feb. 2007, pp. 589–598.
[13] J. A. Kahle, M. N. Day, H. P. Hofstee, C. R. Johns, T. R. Maeurer, and D. Shippy, “Introduction to the cell multiprocessor,” IBM J. Res. Develop., vol. 49, nos. 4–5, pp. 589–604, 2005.
[14] L. Seiler, D. Carmean, E. Spronsh, T. Forsyth, M. Abrash, P. Dubey, S. Jenkins, A. Lake, J. Supperman, R. Cavin, and R. Espasa, “Larabee: A many-core x 86 architecture for visual computing,” ACM Trans. Graph., vol. 27, no. 3, pp. 1–15, 2008.
[15] Y. A. Sadawarte, M. A. Gaikwad, and R. M. Patrikat, ”Comparative study of switching techniques for network-on-chip architecture,” in Proc. Int. Conf. Commun., Comput., Secure (ICCSS), 2011, pp. 243–246.
[16] Y. Xu, J. Zhou, and S. Liu, “Research and analysis of routing algorithms for NoC,” in Proc. 3rd Int. Conf. Comput. Res. Develop., Mar. 2011, pp. 98–102.
[17] A. Agarwal, C. Iskander, and R. Shankar, “Survey of network on chip (NoC) architectures & contributions,” J. Eng., Comput. Archit., vol. 3, no. 1, pp. 21–27, 2009.
[18] G. Ascia, V. Catania, M. Palesi, and D. Patti, “Neighbors-on-path: A new selection strategy for on-chip networks,” in Proc. IEEE/ACM/FIP Workshop Embedded Syst. Real Time Multimedia, Oct. 2006, pp. 79–84.
[19] P. Bogdan and R. Marcheslu, “Non-stationary traffic analysis and its implications on multicore platform design,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 30, no. 4, pp. 508–519, Apr. 2011.
[20] A. Scherrer, A. Fraboulet, and T. Rissett, “Long-range dependence and on-chip processor traffic,” Microprocessors Microsyst., vol. 33, no. 1, pp. 72–80, Feb. 2009.
[21] P. Bogdan and R. Marcheslu, “Statistical physics approaches for network-on-chip traffic characterization,” in Proc. 7th IEEE/ACM Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES+ISSS), Oct. 2009, pp. 461–470.
[22] T. Bjergregaard and S. Mahadevan, “A survey of research and practices of Network-on-chip,” ACM Comput. Surveys, vol. 38, no. 1, p. 1, Jun. 2006.
[23] P. K. Sahu and S. Chattopadhyay, “A survey on application mapping strategies for network-on-chip design,” J. Syst. Archit., vol. 59, no. 1, pp. 60–76, 2013.
[24] A. Abbas, M. Ali, A. Fayyaz, A. Ghosh, A. Kalra, S. U. Khan, M. U. S. Khan, T. De Menezes, S. Pattanayak, A. Sanyal, and S. Usman, “A survey on energy-efficient methodologies and architectures of network-on-chip,” Comput. Electr. Eng., vol. 40, no. 8, pp. 333–347, Nov. 2014.
[25] A. B. Gabis and M. Koudil, “NoC routing protocols—Objective-based classification,” J. Syst. Archit., vol. 66–67, pp. 14–32, May 2016.
[26] Y. Wu, C. Lu, and Y. Chen, “A survey of routing algorithm for mesh network-on-chip,” Frontiers Comput. Sci., vol. 10, no. 4, pp. 591–601, Aug. 2016.
[27] F. Zulkeli, P. Ehkan, M. Warip, N. Y. Phing, and F. Zakaria, “A comparative review of adaptive routing approach for network-on-chip router architecture,” in Proc. Int. Conf. Reliable Inf. Commun. Technol., Cham, Switzerland: Springer, 2017, pp. 247–254.
[28] F. Rad, M. Reshadi, and A. Khademzadeh, “A survey and taxonomy of congestion control mechanisms in wireless network on chip,” J. Syst. Archit., vol. 108, Sep. 2020, Art. no. 101807.
Parallel Programming
T. Rauber and G. Rünger,
J. Duato, S. Yalamanchili, and L. Ni, 
Principles and Practices of Interconnection Networks.
Springer, 2003.
W. J. Dally and B. P. Towles, 
C. J. Glass and L. M. Ni, “The turn model for adaptive routing,” 
K. Ahmad and M. Sethi, “Review of network-on-chip routing algorithms,” 
P. Valencia, “A survey on network-on-chip routing algorithms,” 
T. Rauber and G. Rünger, Parallel Programming. 
J. Duato, S. Yalamanchili, and L. Ni, Interconnection Networks.
San Mateo, CA, USA: Morgan Kaufmann, 2003.
W. J. Dally and B. P. Towles, Principles and Practices of Interconnection Networks.
Amsterdam, The Netherlands: Elsevier, 2004.
T. Rauber and G. Rünger, Parallel Programming. Berlin, Germany: Springer, 2013.
T. Bjerrregaard and S. Mahadevan, “A survey of research and practices of network-on-chip,” 
ACM Comput. Surveys, vol. 38, no. 1, p. 1, Jun. 2006.
K. S. Shim, M. H. Cho, M. Kinsy, T. Wen, M. Li, G. E. Suh, and S. Devadas, “Static virtual channel allocation in oblivious routing,” 
ACM Comput. Surveys, vol. 38, no. 5, pp. 277–356, Sep. 2006.
K. S. Shim, M. H. Cho, M. Kinsy, T. Wen, M. Li, G. E. Suh, and S. Devadas, “Static virtual channel allocation in oblivious routing,” 
ACM Comput. Surveys, vol. 38, no. 5, pp. 277–356, Sep. 2006.
K. S. Shim, M. H. Cho, M. Kinsy, T. Wen, M. Li, G. E. Suh, and S. Devadas, “Static virtual channel allocation in oblivious routing,” 
ACM Comput. Surveys, vol. 38, no. 5, pp. 277–356, Sep. 2006.
J. Yan, “Enhanced global congestion awareness (EGCA) for load balance in networks-on-chip,” in Proc. 7th IEEE/ACM Int. Symp. Netw. Chip (NoCS), Apr. 2013, pp. 1–8.

J. Yan, “Enhanced global congestion awareness (EGCA) for load balance in networks-on-chip,” J. Supercomput., vol. 72, no. 2, pp. 567–587, Feb. 2016, doi: 10.1007/s11227-015-1583-9.

S. Xu, B. Fu, M. Chen, and L. Zhang, “Congestion-aware adaptive routing with quantitative congestion information,” in Proc. IEEE 18th Int. Conf. High Perform. Comput. Commun., IEEE 14th Int. Conf. Smart City, IEEE 2nd Int. Conf. Data Sci. Syst. (HPCC/SmartCity/DSS), Dec. 2016, pp. 216–223.

Y. Gupta and L. Bhargava, “Reinforcement learning based routing for cognitive network on chip,” in Proc. 2nd Int. Conf. Intel. Technol. Competitive Strategies (ICTCS), New York, NY, USA, 2016, p. 55, doi: 10.1145/2905055.2905093.

K. Tatas and C. Chrysostomou, “Adaptive networks-on-chip routing with fuzzy logic control,” in Proc. Euromicro Conf. Digit. Syst. Design (DSD), Aug. 2016, pp. 138–145.

R. Akbar and F. Safaei, “A novel congestion-aware and adaptive routing algorithm in mesh-based networks-on-chip with segmentation,” in Proc. 19th Int. Symp. Comput. Archit. Digit. Syst. (CADS), Dec. 2017, pp. 1–6.

H. C. Touati and F. Boutekkouk, “A weighted minimal fully adaptive congestion aware routing algorithm for network on chip,” in Proc. 1st Int. Conf. Embedded Distrib. Syst. (EDIS), Dec. 2017, pp. 1–5.

Y. Shilova, M. Kavalerov, and I. Bezukladnikov, “Full echo Q-routing with adaptive learning rates: A reinforcement learning approach to network routing,” in Proc. IEEE 14th Int. Conf. Young Res. Elect. Electron. Eng. Conf., Feb. 2016, pp. 341–344.

M. Kavalerov, Y. Shilova, and Y. Likhacheva, “Adaptive Q-routing with random echo and route memory,” in Proc. 26th Conf. Open Innov. Assoc. (FRUCT), Helsinki, Finland, Apr. 2017, p. 20, doi: 10.23919/FRUCT.2017.8071304.

B. Fu and J. Kim, “Footprint: Regulating routing adaptiveness in networks-on-chip,” in Proc. 44th Ann. Int. Symp. Comput. Archit., New York, NY, USA, Jun. 2017, pp. 691–702, doi: 10.1145/3079856.3082049.

M. A. Kinsy, S. Khadka, and M. Isakov, “PreNoc: Neural network based predictive routing for network-on-chip architectures,” in Proc. Great Lakes Symp. VLSI, New York, NY, USA, May 2017, pp. 65–70, doi: 10.1145/3060403.3060406.

S. G. Krishnan, T. Inbarsan, and P. Chitra, “Hybrid adaptive routing algorithm for 2D mesh-on-chip networks,” in Proc. 3rd Int. Conf. Sens. Signal Process. Secur. (ICSSS), May 2017, pp. 284–289.

H. C. Touati and F. Boutekkouk, “FACARS: A novel fully adaptive congestion aware routing scheme for network on chip,” in Proc. 7th Medit. Conf. Embedded Comput. (MECO), Jun. 2018, pp. 1–6.

Z. Li, G. Du, R. Fan, Q. Lu, D. Zhang, and Y. Qiu, “A low-latency adaptive NoC routing scheme based on traffic monitoring,” in Proc. 12th IEEE Int. Conf. Anti-Counterfeiting, Secur. Identif. (ASIC), Nov. 2018, pp. 226–230.

S. Xiao, X. Wang, M. Paleis, A. K. Singh, and T. Mak, “ACDC: An Accuracy- and congestion-aware dynamic traffic control method for networks-on-chip,” in Proc. Design, Autom. Test Eur. Conf. Exifh. (DATE), 2019, pp. 630–633.

F. Farahnakiian, M. Ebrahimi, M. Daneshstahl, P. Liljeberg, and J. Plosila, “Q-learning based congestion-aware routing algorithm for on-chip network,” in Proc. IEEE 2nd Int. Conf. Networked Embedded Syst. Enterprise Appl., Dec. 2011, pp. 1–7.

K. Gaffour et al., “A new congestion-aware routing algorithm in network-on-chip: 2D and 3D comparison,” Int. J. Comput., pp. 1–9, 2019.

R. Fan, G. Du, P. Xu, Z. Li, Y. Song, and D. Zhang, “An adaptive routing scheme based on Q-learning and real-time traffic monitoring for network-on-chip,” in Proc. IEEE 13th Int. Conf. Anti-Counterfeiting, Secur., Identif. (ASIC), Oct. 2019, pp. 244–248.

Y. Luo, M. C. Meyer, X. Jiang, and T. Watanabe, “A hotspot-pattern aware routing algorithm for networks-on-chip,” in Proc. IEEE 13th Int. Symp. Embedded Multicore/Many-Core Syst. Chip (MCSoC), Oct. 2019, pp. 229–235.

Z. Han, M. C. Meyer, X. Jiang, and T. Watanabe, “Low-cost congestion detection mechanism for networks-on-chip,” in Proc. IEEE 13th Int. Symp. Embedded Multicore/Many-Core Syst. Chip (MCSoC), Oct. 2019, pp. 157–163.

F. Bahman, A. Reza, M. Reshadi, and S. Vazifedian, “CACBR: Congestion aware cluster buffer base routing algorithm with minimal cost on NoC,” CCf Trans. High Perform. Comput., vol. 2, no. 3, pp. 1–10, 2020.
[118] J. Fang, D. Zhang, and X. Li, “ParRouting: An efficient area partition-based congestion-aware routing algorithm for NoCs,” *Micromachines*, vol. 11, no. 12, p. 1034, Nov. 2020.

[119] K. Ahmad, M. A. J. Sethi, R. Ullah, I. Ahmed, A. Ullah, N. Jan, and G. M. Karami, “Congestion-aware routing algorithm for NoC using data packets,” *Wireless Commun. Mobile Comput.*, vol. 2021, pp. 1–11, Aug. 2021.

[120] F. Shafiei and V. Sattari-Naeini, “Development of an adaptive multipath routing algorithm by examining the congestion and channel fault of one-hop nodes in network-on-chip,” in *Proc. 8th Int. Conf. Comput. Knowl. Eng. (ICCKE)*, Oct. 2018, pp. 231–236.

[121] A. Jain, V. Laxmi, M. Tripathi, M. S. Gaur, and R. Bishnoi, “TRACK: An algorithm for fault-tolerant, dynamic and scalable 2D mesh network-on-chip reconfiguration,” *Integration*, vol. 72, pp. 92–110, May 2020.

[122] K. Wang and A. Louri, “CURE: A high-performance, low-power, and reliable network-on-chip design using reinforcement learning,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 9, pp. 2125–2138, Sep. 2020.

[123] M. Kumar, V. Laxmi, M. S. Gaur, M. Daneshtalab, M. Ebrahimi, and M. Zwolinski, “Fault tolerant and highly adaptive routing for 2D NoC,” in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst. (DFT)*, Oct. 2014, pp. 104–109.

[124] V. Catania, A. Mineo, S. Monteleone, M. Paleli, and D. Patti, “Noxim: An open, extensible and cycle-accurate network on chip simulator,” in *Proc. IEEE 26th Int. Conf. Appl.-Specific Syst., Archit. Processors (ASAP)*, Jul. 2015, pp. 162–163.

[125] J. Hestness, B. Grot, and S. W. Keckler, “Netrace: Dependency-driven trace-based network-on-chip simulation,” in *Proc. 3rd Int. Workshop Netw. Chip Archit. (NoCAsia)*, 2010, pp. 31–36.

[126] C. Bienia et al., “The PARSEC benchmark suite: Characterization and architectural implications,” in *Proc. 17th Int. Conf. Parallel Archit. Compilation Techn.*, 2008, pp. 72–81.

**RAJENDRA SINGH** received the bachelor’s degree in computer science and engineering (CSE) from Rajasthan University, Jaipur, India, in 2007, and the master’s degree in CSE from Rajasthan Technical University, Jaipur, in 2018. He is currently pursuing the Ph.D. degree with the Department of Computer and Communication Engineering, Manipal University Jaipur. His current research interests include implementation and analysis of efficient and fault-tolerant NoC routing variation in nanoscale technologies.

**MANOJ KUMAR BOHRA** (Senior Member, IEEE) received the bachelor’s degree in computer science and engineering (CSE) from MBM Engineering College, Jodhpur, India, in 2003, and the master’s and Ph.D. degrees in CSE from the MNIT Jaipur, India, in 2011 and 2017, respectively. He is currently a Professor with Manipal University Jaipur. He has more than 15 years of experience in academia and research. His research interests include machine learning algorithms in health care IoT, security, blockchain technology, and networks-on-chip (NoC).

**PRASHANT HEMRAJANI** received the bachelor’s degree in information technology from BMIT, Sitapura, Jaipur, India, in 2009, and the master’s degree in computer science and engineering from RIET, Bhankrota, Jaipur, in 2014. He is currently an Assistant Professor with Manipal University Jaipur. He has more than ten years of experience in academia and research. His research interests include networking and machine learning.

**ANSHUMAN KALLA** received the Engineering degree from Government Engineering College Bikaner, in 2004, the Master of Science degree in telecommunications and wireless networking from ISEP, Paris, France, in 2008, the master’s degree from the University of Nice Sophia Antipolis, France, in 2011, and the Ph.D. degree, in 2017. He is currently working as a Professor with the CGPIT, Uka Tarsadia University, India. He has more than 12 years of teaching and research experience. He has delivered invited sessions and talks during various short term courses, faculty development programs, workshops, and conferences. His research interests include blockchain, next generation networks-5G and 6G, the Internet of Things (IoT), information centric networking (ICN), and software-defined networking (SDN). More information can be found at: https://sites.google.com/site/kallanshuman/.

**DEVESHI PALLAVI BHATT** received the master’s degree in computer applications from MDU, Rohtak, Haryana, India, in 2006, the master’s degree in computer science and engineering from Jannagath University, Jaipur, India, and the Ph.D. degree in CSE from Banasthali Vidyapith, India, in 2016. She is currently a Professor and the Head of the Department of Computer Applications, Manipal University Jaipur. She has more than 15 years of experience in academia and research. Her research interest includes wireless networks.

**NITIN PUROHIT** received the degree in computer science and engineering from Rajasthan University, Jaipur, Rajasthan, India, and the Master of Technology degree in computer science and engineering from Rajasthan Technical University, Rajasthan. He is currently working as an Associate Professor with the Department of Computer Science, School of Computational and Natural Sciences, Kebri Dehar University, Ethiopia. He is in teaching profession for more than ten years and has international experience along with experience in institutes of national importance, India. He has presented number of papers in national and international journals, conference and symposiums, and reviewed several papers. His main research interests include high-performance computing, data science, and information security.

**MASOUD DANESHTALAB** is currently a Full Professor with Mälardalen University, Sweden, and an Adjunct Professor with TallTech, Estonia, and leads the Heterogeneous System Research Group. He has published over 200 refereed papers. His research interests include HW/SW co-design, reliability, and deep learning acceleration. He is on the Euromicro Board of Directors and an Editor of the *MICPRO* journal.