electronics

MDPI

Article

RLARA: A TSV-Aware Reinforcement Learning Assisted Fault-Tolerant Routing Algorithm for 3D Network-on-Chip

Jiajia Jiao *, Ruirui Shen, Lujian Chen, Jin Liu and Dezhi Han

College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China; 202130310193@stu.shmtu.edu.cn (R.S.); 202030310156@stu.shmtu.edu.cn (L.C.); jinliu@shmtu.edu.cn (J.L.); dzhan@shmtu.edu.cn (D.H.)

* Correspondence: jiaojiajia@shmtu.edu.cn

Abstract: A three-dimensional Network-on-Chip (3D NoC) equips modern multicore processors with good scalability, a small area, and high performance using vertical through-silicon vias (TSV). However, the failure rate of TSV, which is higher than that of horizontal links, causes unpredictable topology variations and requires adaptive routing algorithms to select the available paths dynamically. Most works have aimed at the congestion control for TSV partially 3D NoCs to bypass the TSV reliability issue, while others have focused on the fault tolerance in TSV fully connected 3D NoCs and ignored the performance degradation. In order to adequately improve reliability and performance in TSV fully connected 3D NoC architectures, we propose a TSV-aware Reinforcement Learning Assisted Routing Algorithm (RLARA) for fault-tolerant 3D NoCs. The proposed method can take advantage of both the high throughput of fully connected TSVs and the cost-effective fault tolerance of partially connected TSVs using periodically updated TSV-aware Q table of reinforcement learning. RLARA makes the distributed routing decision with the lowest TSV utilization to avoid the overheating of the TSVs and mitigate the reliability problem. Furthermore, the K-means clustering algorithm is further adopted to compress the routing table of RLARA by exploiting the routing information similarity. To alleviate the inherent deadlock issue of adaptive routing algorithms, the link Q-value from reinforcement learning is combined with the router status based in buffer utilization to predict the congestion and enable RLARA to perform best even under a high traffic load. The experimental results of the ablation study on simulator Garnet 2.0 verify the effectiveness of our proposed RLARA under different fault models, which can perform better than the latest 3D NoC routing algorithms, with up to a 9.04% lower average delay and 8.58% higher successful delivered rate.

Keywords: deadlock mitigation; fault tolerance; K-means clustering; through silicon vias; 3D network-on-chip; reinforcement learning

Citation: Jiao, J.; Shen, R.; Chen, L.; Liu, J.; Han, D. RLARA: A TSV-Aware Reinforcement Learning Assisted Fault-Tolerant Routing Algorithm for 3D Network-on-Chip. *Electronics* **2023**, *12*, 4867. https://doi.org/10.3390/electronics12234867

Academic Editor: Marcin Witczak

Received: 21 October 2023
Revised: 17 November 2023
Accepted: 20 November 2023
Published: 2 December 2023

1. Introduction

With advances in the scaling of node technology down to 5 nm, an increasing number of processors in multicore architectures can be integrated into a single chip. Network-on-Chips (NoCs) are preferred in multicore processors because of their low communication latency, high network bandwidth, and good scalability [1]. Using three-dimensional (3D) stacking technology, 3D NoCs can achieve lower latency and lower area consumption [2,3]. The 3D mesh topology is widely used in academic research and industrial products because it is symmetric and regular, with a standard chip shape [4]. Silicon vias (TSVs) are adopted to implement vertical links and connect the multiple layers in a 3D mesh, since they can achieve a lower latency, minor power consumption, a smaller area, and a higher density than conventional links [5]. Therefore, TSV-connected 3D NoCs are becoming increasingly popular in modern processor design.

The new reliability issue in 3D NoCs results from the high susceptibility of TSVs to high failure [6]. This is caused by the generated gaps and deviations that occur during

the TSV fabrication process and is determined by the inherent characteristics of necessary materials with a wide range of chemical and mechanical properties. More importantly, the highly frequent usage of TSVs, even though it provides a lower latency, also causes severe overheating and a higher failure rate [7]. The critical challenges of TSVs in 3D NoCs include two aspects: (1) higher failure rate because of heating problems and low yield [8]; (2) higher area overhead of bigger TSV pads than conventional wires [9,10]. Therefore, designing adaptive routing algorithms for fault-tolerant 3D NoCs is significant in order to guarantee communication quality.

Most existing works on routing algorithms have aimed at the congestion control for TSV partially 3D NoCs to bypass the TSV reliability issue. For example, the adaptive routing algorithms LEAD [11], Advertiser Elevator [12], Reflect3D [13], TCAR [14], AdEle+ [15], and traffic-aware-based TAFT [16] have been proposed to reduce area and alleviate the energy consumption of TSVs. The others have used 3D NoCs with fully connected TSVs due to their high efficiency. It has been proven that an adaptive, fault-tolerant, and congestion-aware (AFTC) routing algorithm [17], deadlock-free HLAFT [18], and a fault-tolerant routing algorithm using the Hamiltonian path strategy [19] can utilize more vertical TSVs in 3D NoCs for high communication performance. The former aims at performance improvements and energy optimization with limited TSVs, while the latter focuses on the reliability mitigation. Therefore, it is critical to balance the guaranteed fault tolerance and performance degradation in 3D NoC routing algorithms. To address the TSV reliability issue and performance loss jointly, we propose a TSV-aware Reinforcement Learning Assisted fault tolerant Routing Algorithm (RLARA) for varying 3D NoCs.

A comparison between the proposed RLAR and the related works is presented in Table 1. Though the latest routing algorithms have used some turn models, virtual channels, and graph partition to optimize the communication performance, their adaptiveness is limited for the varying TSV-connected 3D NoCs. It is observed that the reinforcement learning technique in our proposed RLARA has been used for effective routing algorithms [20,21] to select routing paths adaptively. It can take advantage of both the lower failure rate of partially connected TSVs and the high efficiency of fully connected TSVs. More importantly, RLARA can periodically update the 3D NoC routing decisions using a reinforcement learning method and adapt to topology changes with fault horizontal links or TSVs. The main contributions are as follows:

- We design a novel fault-tolerant routing algorithm RLARA, which uses Q-learning to handle the changing 3D NoC topology due to faults occurrence, and explores the higher failure rates of vertical TSVs in path selection for better communication performance;
- We develop a K-means clustering mechanism to compress the routing table by exploiting the routing information similarity. The degradation of the successfully delivered rate is just 1.57%, while the storage space for a routing Q-Table is reduced by 70%;
- We implement a mixed-status load balance mechanism to mitigate the deadlock issue in our adaptive RALAR. The link Q-value and the router buffer are combined to predict the congestion and decide on the possible routing paths. This scheme helps RALAR to perform better than the latest works, even under the high traffic load of 3D NoCs;
- We conducted many experiments on the open-source NoC simulator Garnet. The experimental results of varying network sizes and fault models prove that our proposed RLARA achieves better communication performance of up to a 9.04% lower latency and an 8.58% higher successful delivery rate than state-of-the-art routing algorithms on 3D NoCs.

The remainder of this work is organized as follows. Section 2 introduces the background and related work. Section 3 details the proposed method, while the results and analysis are described in Section 4. Finally, the conclusion is summarized in Section 5.

Table 1. Routing algorithm comparison for 3D NoC.

Algorithms	Characteristics	Limitations
LEAD [11] (2018)	+Random or shortest path TSV +Deadlock mitigation by splitting the topology	-Just suitable for TSV partially connected 3D NoC
Advertiser Elevator [12] (2017)	+Divide topology into three sections +Increases virtual channels to alleviate congestion	-Just suitable for TSV partially connected 3D NoC
TAFT [16] (2019)	+Selects routing using traffic density +Simple implementation	-Packet cycling and suboptimal routes
Q-thermal [21] (2020)	+Balances thermal distribution across different layers of the chip using Q-learning technique	-More area and power consuption -No fault tolerance
Reflect3D [13] (2021)	+Provides adaptive fault-tolerant routing +Ensuring packet delivery with at least one healthy TSV connecting all layers	-Just suitable for TSV partially connected 3D NoC
TCAR [14] (2022)	+Optimizes thermal distribution based on the allocation of thermal and congestion weights +Optimizes path selection by considering network congestion status	-Just suitable for TSV partially connected 3D NoC
AFTC [17] (2023)	+ Introducing 12 turn models to effectively avoid deadlock +Using a three-dimensional diagonal region division method for better adaptability +Reducing turn steps during fault situations +Setting different congestion thresholds to balance traffic	-Lack of realistic applications mapped into 3D NoCs for comprehensive verfication -No TSV consideration
AdEle+ [15] (2023)	+Dynamically manages traffic congestion on elevators +AdEle+ switches to a distance-based selection in low congestion +AdEle+ is lightweight, adaptable to runtime traffic loads, and selects the best elevator during runtime	-Just suitable for TSV partially connected 3D NoC
Our RLARA	+Dynamic routing selection using reinforcement learning for both local and global information +K-means clustering to shrink routing table addresses +TSV aware routing selection +Suitable for fully or partially TSV-connected 3D NoC	-Lack of realistic applications mapped into 3D NoCs for comprehensive verification

2. Background and Related Work

This section briefly introduces the basic fault 3D NoC architecture, related fault-tolerant routing algorithms, and K-means clustering.

2.1. Fault 3D NoC Architecture with Fully Connected TSVs

Our work considers the often-used 3D Mesh with fully connected TSVs as the basic 3D NoC architecture, as shown in Figure 1a. Each internal router has six neighbors in the west, east, north, south, up and down, and one local direction, by which it connects with the IP core, as shown in Figure 1b. Its router has seven input ports, a seven by seven crossbar and seven output ports to pipeline the injected packets via the popular wormhole flow control mechanism. The corner or boundary routers have fewer neighbors.

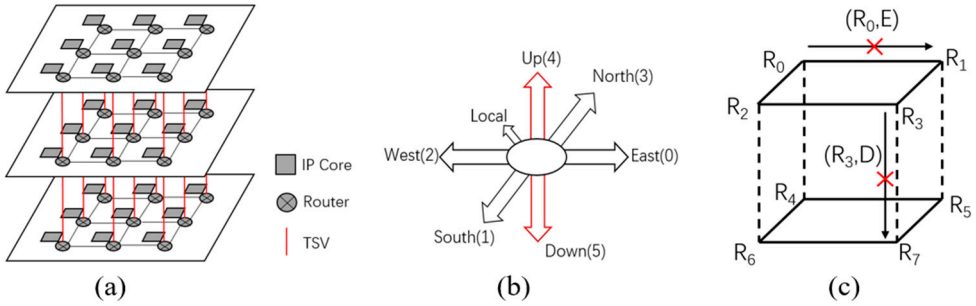


Figure 1. TSVs-based fault 3D NoC architecture. (**a**) The 3D Mesh with vertical TSVs; (**b**) internal routers with seven directions; (**c**) link faults.

The faults are injected randomly in the horizontal links or vertical TSVs at a specific fault injection rate between different routers as is shown in Figure 1c. Algorithm 1 details the fault model in a 3D NoC with three inputs and one output. Using lines 2 to 9, the faults are randomly injected into the horizontal links at a lower fault rate, while the similar faults are injected into TSVs at a higher fault rate in line 10. We assume that the topology is a strongly connected graph for a standard routing algorithm in line 5. Otherwise, the routing procedure is interrupted, and the complete statistics are unavailable. Moreover, the multiple fault injection rates are configured to model the varying and realistic scenarios of low, medium, and high loads, respectively. To dynamically bypass the fault horizontal links or TSVs, the fault-tolerant 3D NoC routing algorithm should capture the routing information and select from the candidate routing paths.

Algorithm 1: Fault model

Input: fault_injection_rate, total_num_link, TSV_fault_ratio
Output: fault_links
1. Initialize fault_link_num to 0 and fault_links as an empty array
2. while fault_link_num < fault injection rate * total_link_count * (1- TSV_fault_ratio)
3. Currently ← a random router number
4. dir ← a random direction
5. if <current, dir> not in fault_links && new topology is strongly connected graph
6. fault_links.append(<current, dir>)
7. fault_link_num + = 1
8. end if
9. end while//inject a fault into the horizontal links with a lower fault ratio
10. repeat the above loop and inject faults into the TSV links with a higher fault ratio
11. return fault_links

2.2. Fault Tolerant Routing Algorithm for 3D NoC

The 3D NoC routing algorithms have been widely studied to satisfy different communication requirements. LA-XYZ [22], M-XYZ [23], and DyXYZ [24] are proposed for use in regular 3D Mesh architectures. They can address the 3D NoC deadlock problem and the congestion challenge well, but cannot handle the increasing faults in TSVs.

The reinforcement learning technique is good for optimizing fault-tolerant routing algorithms in 3D NoCs [20]. Q-learning, as one model-free reinforcement learning method, is more suitable for fault routing algorithm design from a cost-effective perspective. The collected fault information can be stored in a routing Q-Table, which is updated periodically for dynamic path selection. Some 3D NoC routing solutions use Q-learning to alleviate congestion [20] and do the thermal management [21]. Inspired by the Q-learning based fault-tolerant algorithm for 2D NoCs [25,26], we try to design a 3D NoC fault-tolerant routing algorithm RLARA.

When the fault links occur in the TSVs, the 3D NoC topology with fully connected TSVs becomes a partially vertical TSVs-based 3D NoC. Therefore, their routing algorithms

can be adaptive for a fault-tolerant 3D NoC with partial TSVs. An adaptive routing algorithm LEAD [11] is proposed based on the queuing theory for fast analytical performance evaluation. Based on the well-known distributed routing algorithm Elevator First [27], Advertiser Elevator [12] tracks the vertical links failure and re-routes the packets around using the collected message in 3D NoC with partially vertical TSVs. TAFT [16] considers the faulty links to the maximum traffic state and avoids using them. Meanwhile, two or more virtual channels are used [28–30], or some constraints are set on the vertical links to guarantee deadlock-free routing algorithms [31].

Unlike these works, our proposed method RLARA uses TSV-aware information to design a simple and effective routing algorithm to improve the communication performance and alleviate the deadlock problem.

2.3. K-Means in 3D NoC Routing

Compared with hierarchical clustering and density-based clustering, partition-based clustering involves lightweight computing and a fast speed. It has been used in some 3D NoC routing algorithm designs, such as the K-means multicast routing algorithm [32] and our previously designed KARL routing algorithm [33].

To characterize the packet transmission and make good routing decisions for a fault-tolerant 3D NoC, the Q-Table used in the reinforcement learning method records the probabilities of all directions that should go from a current router to a destination router. If the total number of routers is N, the size of the Q-Table will be bigger than N^2 and becomes very large for a 3D NoC. Therefore, it is necessary for K-means clustering to exploit the routing information similarity and compress the Q-Table very well.

3. Proposed TSV-Aware Routing Algorithm RLARA

3.1. Motivation

The higher failure rate of TSVs than that of horizontal links means that more routing algorithms make good use of partial TSVs [11,12,14–16] to avoid the heat problem. However, their adaptiveness and the potential deadlock issues are challenging. Therefore, some researchers try to use turn models, region division and congestion threshold control to further exploit the efficiency of more TSVs in 3D NoCs [17,34]. Without being aware of the varying fault links, global network information and local communication status, these routing algorithms have shown limited adaptiveness. Therefore, our proposed routing algorithm RLARA makes good use of the reinforcement learning technique, K-means clustering method and TSV-aware design to achieve good adaptiveness, high communication performance and guaranteed fault tolerance in 3D NoCs.

3.2. Overall TSV-Aware Reinforcement Learning Routing Algorithm

RLARA is a reinforcement learning-assisted routing algorithm for fault-tolerant 3D NoCs. It can make the routing decision adaptively by considering vertical TSVs utilization and the mixed faults status. A K-means clustering method is used to achieve excellent trade-off between performance degradation and area overhead.

The reinforcement learning method takes advantage of an agent to maximize its benefits through interactive actions in a complex and uncertain environment [35]. As an off-policy reinforcement learning method, Q-Learning is preferred for designing fault-tolerant routing algorithms because of its random exploration and experience reuse capacity. Random exploration enables one to traverse all the candidate routing paths as soon as possible so that the agent can track the links' and routers' situations in the changing 3D NoC topology. It helps RLARA to select a TSV-aware shortest routing path with minimal TSVs from all candidates. Reusing experience enables RLARA to use the collected information from Q-Table and adapt to new fault models dynamically.

RLARA uses the Q-learning method to determine the action of routing the next hop based on the Q-Table update, as Figure 2 shows. A large Q-Table is required to record the Q-value and the product of the total number of states. Therefore, the total number

of routing actions determines the Q-Table size. For example, a 4 × 4 × 4 3D NoC has 64 routers, and thereby 64 × 64 states, as shown in Figure 3, while there are six actions with adjacent directions (E, S, W, N, U, D) in a 3D NoC. Therefore, the Q-Table's size is $6N^2$ for the 3D NoC when the total number of routers is N. If some actions are forbidden for the corner or border routers with fewer adjacent neighbors, the actions are invalid and will be done again. Q-Values in the Q-Table are initialized as 0 and updated by the Q-learning training procedure in Algorithm 2.

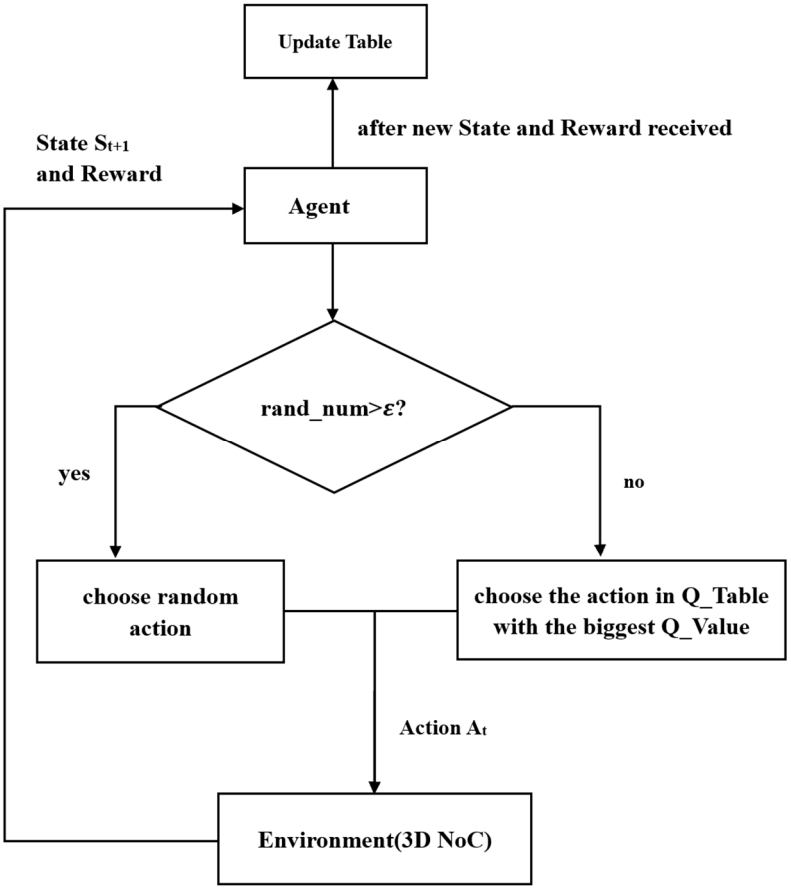


Figure 2. Relationship between agent and environment in RLARA.

States	E	S	W	N	U	D
Q(R0,R0)	0	0	0	0	0	0
Q(R0,R1)	0	0	0	0	0	0
.....						
Q(R0,R63)	0	0	0	0	0	0
Q(R1, R0)	0	0	0	0	0	0
.....						
Q(R63, R63)	0	0	0	0	0	0

Q_Values

Figure 3. Initial 64 × 64 states in the Q-Table of RLARA under 4 × 4 × 4 3D NoC.

Algorithm 2 details the complete Q-Table training procedure of RLARA. To achieve a balance between the TSV's high failure rate and the shortest path using TSVs, RLARA calculates the Q-values of available routes and chooses the routing path with minor TSV

vertical links to transmit packets. Initialization is performed from line 1 to line 6. The ε -greedy strategy uses the value of ε to decide on the exploration strategy or the exploitation strategy using line 7 to line 12. The agent will randomly choose an action At if the exploration strategy is adopted. Otherwise, if using the exploitation strategy, At will be the corresponding action whereat the Q-value is the biggest. The environment will return a new state St + 1 and reward R to the agent after it receives this action At.

Algorithm 2: Q-Table training procedure of RLARA

Input: 3D Mesh topology environment (env), Q-Table (old Q-Table before training), learning rate α, discount factor γ, and the strategy rate ε for greedy strategy.
Output: Q-Table (new compressed Q-Table after training)

```
1.  initialize the candidate routes all_routes as empty
2.  for i = 0 to training_round
3.      initialize the current route as empty
4.      while done == True
5.          St ← [start, end]
6.          while env.is_legal(St, At)
7.              if a random number > ε
8.                  select a random action At
9.              else //mixed status load balance scheme
10.                 At ← action of max(Q-table[S, :])
11.         end if
12.         end while
13.         1append [St, At] to route
14.         St+1, R ← env.step(St, At)
15.         if St+1 is a destination
16.             done ← True
17.             append [St+1, 0] to route
18.             append route to all_routes
19.     end if
20.     Q-table[St, A] ← Qtable[St, A] + α × (R + γ× max(Qtable[St+1, :])–Q-table[St, A])
21.   end while
22. end for
23. best_route ← select the route with the least TSV vertical links from all_routes
24. K-means(Q-Table)//use K-Means to compress Q-Table
```

Then, RLARA saves the routing states in a list "route" after choosing an action from line 13 to line 17. Once the destination has been reached, RLARA appends a complete "route" to the candidate routes "all_route" in line 18. The agent will update the Q-Table, and the Q-values in the Q-Table are recalculated in line 20 using the following formula [36]:

$$Q(St,\ At) = Q(Si,\ Aj) + \alpha \times [R + \gamma \times max_A Q(St+1,\ At) - Q(St,\ At)] \quad (1)$$

where R is the feedback reward of a new topology environment according to the selected routing action At. In this paper, the reward will take a positive value if the routers are reachable. On the contrary, it will be a negative number if the routers are unreachable and connected to faulty links.

At the end of this training procedure, RLARA selects the best_route from all candidates in "all_route" in Line 23. Additionally, the K-means clustering method can be used to compress the Q-Table effectively in line 24.

To illustrate this procedure, we give a simple example of the source R0, and the destination is R7 in Figure 1c. α is set to 0.1 and γ is set to 0.9. The Reward will be 10 upon reaching the destination and otherwise it will be 0 or -10. At first, all the values in the Q-Table are initialized to 0, and the related Q-Values in the table for the paths from R0 to R7 are also zero. Let us assume it chooses to forward the packet to R1, i.e., the action is "E". Upon updating the Q-Table, Max(Q((R1, R7), A)) = 0, and Q((R0, R7), E) = 0. Since R1 is not the destination, the Reward is 0. Then, we calculate the new value using Formula (1).

The result remains 0, signifying no learning has occurred. This situation persists until any exploration reaches the destination. When R3 chooses to transmit the data packet downward during exploration, the Reward is set to 10 for this action. When updating the table, Max(Q((R7, R7), A)) = 0 and Q((R3, R7), D) = 0. We calculate the new value again by Q((R3, R7), D) = 0 + 0.1 × (10 + 0.9 × 0 − 0) equal to 1. This result makes the packet more inclined to choose "Down" as the output direction when its next hop reaches R3. In the following rounds, if other routers send packets to R3, the Q-Table will continue to be updated. For example, assuming that at R1 it sends a data packet to R3, it is determined that when Max(Q((R3, R7), A)) = 1, Q((R1, R7), S) = 0, and Reward = 0, the New Q-Value will be 0.09. Even with a modest increase of 0.09, it makes the packet more likely to be sent to R3 the next time it reaches R1. Similarly, R0 transmitting packets to R1 also updates the Q-Table. Then, Q((R0, R7), S) = 0 + 0.1 × (0 + 0.9 × 0.09 − 0) = 0.0081. This continual accumulation allows the Q-Table to be continually refined, eventually approaching a better path. In the scenario where the path from R3 to R7 experiences a malfunction, rendering data transmission impossible, and assuming R3 selects Down as the output direction, causing the data to be routed through a faulty path, let us assume the obtained Reward is −10. In such a situation, after updating the Q-Table results in Max(Q((R7, R7), A)) = 0, we can get Q((R3, R7), D) equal to 0.81. Substituting this into the formula, Q((R3, R7), D) = 0.81 + 0.1 × (−10 + 0.9 × 0 − 0.81) = −0.271. Through these iterations, the data packet exhibits an even greater reluctance to select this specific path as the output direction.

3.3. K-Means Clustering-Based Routing Table Compression

RLARA further exploits the similarity of routing information to shrink the large Q-Table using the K-means clustering method in Algorithm 3.

The value of K and the original Q-Table are the inputs. To initialize K clustering centers, K states A = [A1, A2, . . ., Ak] are randomly selected from all states in the trained Q-Table and saved from line 1 to line 2. For each state in Q-Table, the Euclidean distance between it and each cluster center is calculated by Equation (2), where E_{Ai} denotes the east Q-Value of Ai in a cluster center, and E_{Sj} represents the east Q-value of corresponding State Sj in the Q-Table. Similarly, the other directions can be defined.

$$d(Ai, Sj) = \sqrt{\begin{array}{c}(E_{Ai} - E_{Sj})^2 + (S_{Ai} - S_{Sj})^2 + (W_{Ai} - W_{Sj})^2 + (N_{Ai} - N_{Sj})^2 \\ +(U_{Ai} - U_{Sj})^2 + (D_{Ai} - D_{Sj})^2\end{array}} \quad (2)$$

Using lines 3 to 13, all the states are classified into one of the K clusters based on the minimal Euclidean distance. Then, the average Euclidean distance is calculated using Equation (3) to select the new center point of each cluster from line 14 to line 16.

$$E_{Ai} = \frac{1}{n}\left(\sum_{x=1}^{n} E_{ix}\right) \quad (3)$$

This procedure is repeated, and K-means are returned unless the results of all cluster centers show no significant change between line 17 and line 19. In summary, RLARA can use K-means clustering to compress the large Q-Table to ensure its scalability in multicore processors. More results are shown in Section 4.3.

3.4. Deadlock Mitigation Using Mixed Status Load Balance

More virtual channels (VCs) or limited link usages are often used to guarantee deadlock freedom and avoid communication quality degradation. In this paper, we use a load balance scheme to mitigate the deadlock issue in an adaptive routing algorithm. This scheme uses both the link Q-Value from Q-learning and the router buffer to decide the next hop.

Wr denotes the unoccupied buffer utilization in a router port, while the product of Wr and the Q-Value are used to select routing direction from all candidates. For example, the Q-Value of a link is 0.57 in the east and 0.29 for the south. If only considering the Q-Value, east is better. However, the unoccupied buffer utilization is 22% for the east and 64% in the south. Comparing the joint value 0.1254 (0.57 $\times$ 22%) with 0.1856 (0.29 $\times$ 64%), the south is better according to the mixed status of the router and link together.

The complete deadlock mitigation scheme is detailed in Algorithm 4. It helps us to select the best direction for the next hop routing. The algorithm uses a variable ε to control whether to use an exploit strategy or an exploratory strategy in line 1. After obtaining the unoccupied buffer utilization of the router and the corresponding link Q-value from the Q-Table, their product is calculated from line 3 to line 7. The direction with the highest product is selected as the final decision.

Algorithm 3: K-Means clustering algorithm

Input: Qtable, K clusters
Output: kmeans (after clustering), index

```
1.  kmeans = random K States in Qtable
2.  old_kmeans = kmeans
3.      for count = 0 to N // N is the upper bound of repeated procedure
4.          for i = 0 to Qtable's size // traverse all states
5.              min_dis = inf
6.              for j = 0 to K // traverse all K clusters
7.                  dis = d(Qtable[i], kmeans[j]) // using Equation (2)
8.                  if dis < min_dis
9.                      min_dis = dis
10.                 index[i] = j
    // index shows which group is the state in
11.             end if
12.         end for
13.     end for
14.     for each row in kmeans
15.         update kmeans with the average Q-Values
    of all the states in a group
16.     end for
17.     if old_kmeans == kmeans // no change and stop the procedure
18.         break
19.     end if
20.     end for
```

Algorithm 4: Deadlock mitigation via mixed status load balance scheme

Input: Qtable, state, ε decides whether exploration or exploitation strategy.
Output: maximum

```
1.  if rd.randint(1, 100)/100 < ε then
2.      for direction = 1 to Max_ports
3.          Wr ← get_next_buffer(state, direction)
4.          if buffer * Qtable[state][i] > max_value then
5.              maximum := i
6.              max_value := C * Qtable[state][i]
7.          end if
8.      end for
9.  else
10.     maximum := rd.randint(0, Max_ports-1)
11. end if
```

4. Experiments and Analysis

4.1. Experiments Configuration

The experiments were performed in the open-source NoC simulator Garnet 2.0 [37]. To reflect the higher failure rate of TSV under realistic scenarios, the ratio of fault injection into TSV and conventional links is set to 4:1 [5]. The Q-learning configuration refers to the work in [38]. To verify the proposed RLARA, we use two often-used metrics—average latency and successfully delivered rate—and compare our work with five methods, which are the ideal routing algorithm [37], LEAD [11] and Advertiser Elevator denoted as adv_ele [12], TAFT [16], and Elevator first denoted as ele_first [27] for 3D NoC with different network size. The typical $3 \times 3 \times 3$ and $4 \times 4 \times 4$ topologies [15,34] are selected to verify the effectiveness of the proposed routing algorithm. The configuration with more parameters is listed in Table 2, which includes the related parameters for network and reinforcement learning.

Table 2. Experimental parameters configuration.

Experimental Parameters	Values
Network size	$3 \times 3 \times 3$, $4 \times 4 \times 4$
Fault injection rate	5%, 10%, 15%, 20%
Packet injection rate(Packet/cycle/node)	0.01, 0.02, 0.03, 0.04, 0.05, 0.06
Compression rate	30%, 50%, 70%, 90%
The ratio of inject fault into TSV and conventional links	4:1
Fault injection mode	Random
Packet size	64 bits
Synthetic traffic	Uniform random
Link width	32 bits
The number of Simulation cycles	50,000
Learning rate α for reinforcement learning	0.01
Discount factor γ for reinforcement learning	1

4.2. TSV-Aware Routing Impacts on Communication Performance

4.2.1. Network Performance: Average Latency and Successfully Delivered Rate

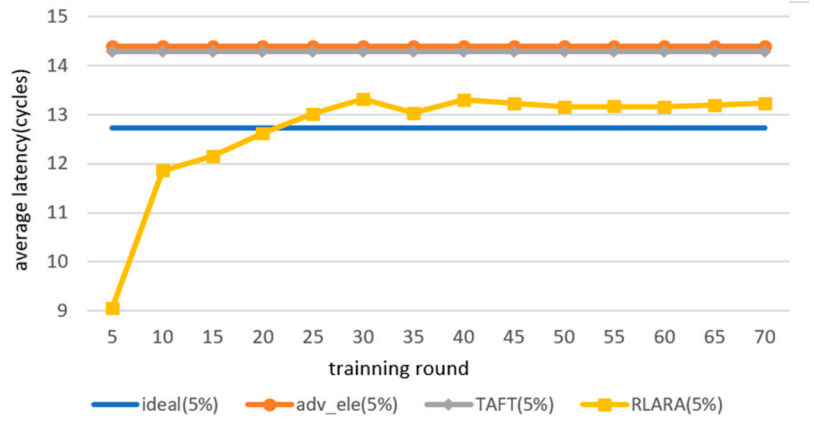
The average latency and successfully delivered rate are compared among different routing algorithms for a $4 \times 4 \times 4$ 3D NoC under a 5% fault link injection in Figure 4. RLARA performs similarly to the ideal case—better than TAFT [16] and adv_ele [12] in the stable stage but worse in the early stage. The reason is that the extra exploration of Q-Table training starts from an initial state in the early stage and searches for the best routing path later. Here, training all the source routers and the destination routers one time is called a training round. Because RLARA cannot search the best routing path in the early training stage, we only record the average latency and successful delivery rate starting from five training rounds. The performance metrics are recalculated every five training rounds. After 25 to 30 training rounds, the successfully delivered rate increases quickly and enters a stable stage, as shown in Figure 4b. The lower average latency in the early stage mainly results from the lower packet transmission shown in Figure 4a.

Then, the average latency will gradually decrease in the middle stage because the training is imperfect and many packets do not always take the shortest route.

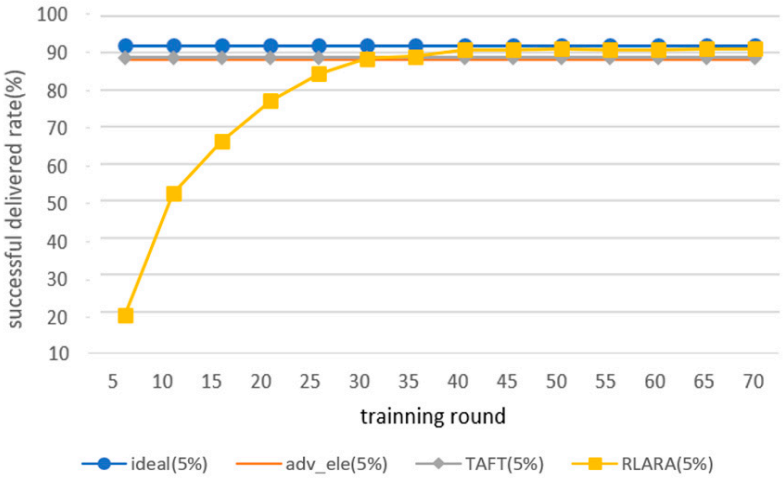
Finally, all the packets can find the best route to the destination router when the training enters the saturation region, where the optimal paths are selected and very close to the ideal case. Unlike RLARA, TAFT and adv_ele give the fixed routing decisions without an extra training process.

The improvement of RLARA over adv_ele and TAFT routing algorithms is detailed in Table 3. The average latency and the successful delivered rate are all improved for $4 \times 4 \times 4$ Mesh under varying fault injection rates of 5%, 10%, 15% and 20%. RLARA can decrease the average latency by 9.04% over adv_ele under a low fault injection rate of 5%. The improved successful delivery rate of RLARA over TAFT increases up to 8.58%.

Therefore, the proposed RLARA achieves a lower latency and a higher successful delivery rate than the TAFT and adv_ele for 4 × 4 × 4 Mesh under different fault injection rates.



(**a**)



(**b**)

Figure 4. Performance comparison for the various routing algorithms for 4 × 4 × 4 3D NoC under a 5% fault injection rate. (**a**) Average latency; (**b**) successful delivery rate. Compared with the TAFT and adv_ele, the proposed RLARA is closer to the ideal case after initial training, with lower average latency and higher successful delivered rate.

Table 3. Improvement of RLARA with different fault injection rates under 4 × 4 × 4 3D NoC.

Varying Fault Injection Rate in 4 × 4 × 4 3D NoC		**5%**	**10%**	**15%**	**20%**
Lower average latency	RLARA—adv_ele	9.04%	7.99%	7.40%	7.63%
	RLARA—TAFT	8.21%	7.51%	6.53%	7.34%
Higher successful delivered rate	RLARA—adv_ele	2.95%	4.56%	4.82%	7.77%
	RLARA—TAFT	2.41%	4.83%	4.56%	8.58%

For a 3 × 3 × 3 network, Table 4 shows the similar results and lower performance improvements of RLARA over adv_ele and TAFT—up to an 8.48% lower average latency and 6.21% higher successful delivered rate. The larger the network is, the higher the performance benefits of RLARA. This is because the large topology provides more TSVs and route candidates for RLARA.

Table 4. Improvement of RLARA with different fault injection rates under 3 × 3 × 3 3D NoC.

Varying Fault Injection Rate in 3 × 3 × 3 3D NoC		5%	10%	15%	20%
Lower average latency	RLARA—adv_ele	8.48%	8.37%	5.68%	3.57%
	RLARA—TAFT	5.45%	4.64%	5.08%	5.36%
Higher successful delivered rate	RLARA—adv_ele	0.62%	0.62%	1.25%	6.21%
	RLARA—TAFT	0.62%	3.11%	4.94%	5.86%

In a word, RLARA performs better than the state-of-the-art under varying fault injection rates and network sizes. It displays a minimal performance gap to the ideal routing algorithm.

4.2.2. TSV Usage

TSV usage balance is a critical factor for RLARA. Comparisons of the average numbers of used TSVs at different fault injection rates (5% and 15%) and network sizes (3 × 3 × 3 and 4 × 4 × 4) between the proposed RLARA and TAFT [16] are given in Figure 5. This part excludes adv_ele because it is partially TSV connected and cannot deal with the faults in a specific dimension. The average TSV usage is measured for 100 different randomly generated fault models.

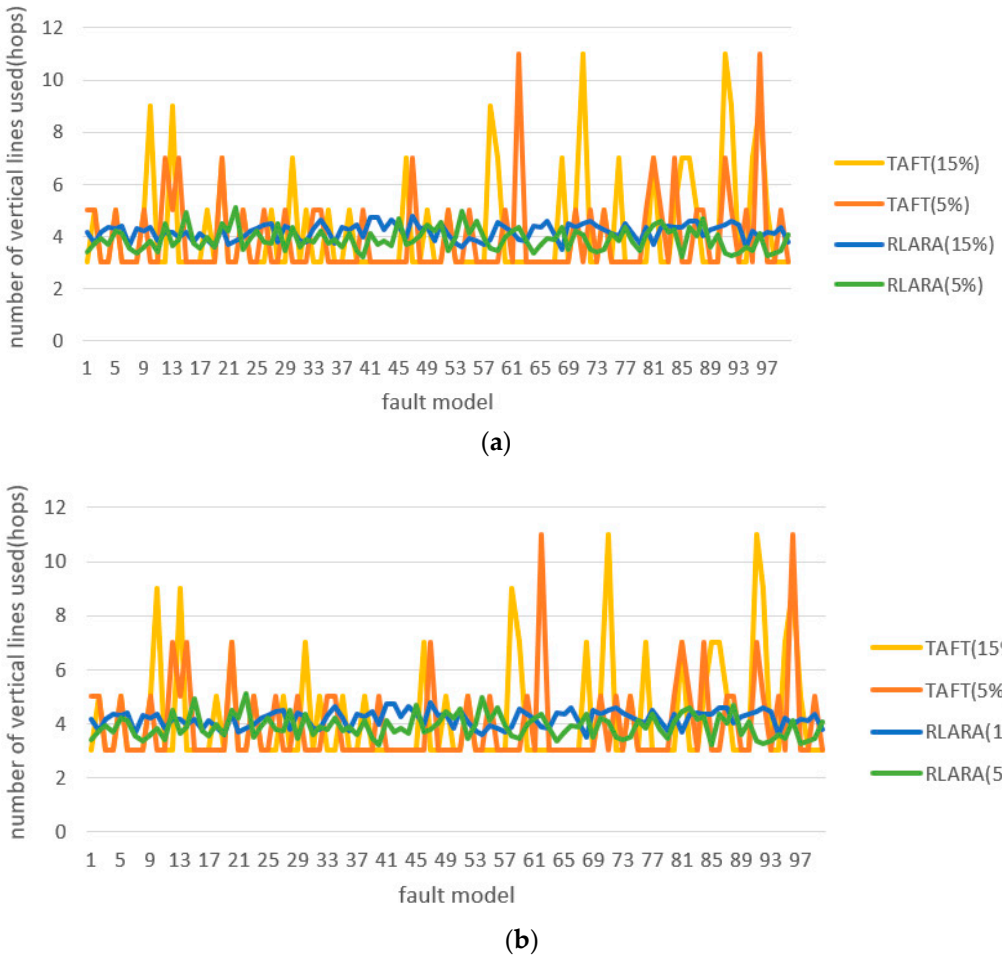


Figure 5. The vertical TSV usage comparison under varying network sizes and fault injection rates: (**a**) 3 × 3 × 3 NoC; (**b**) 4 × 4 × 4 NoC. The network size has little impact on the TSV usage while the higher injection rate leads to higher utilization. Compared with TAFT, the proposed RLARA can achieve stable and balanced TSV usage and thus avoid overheating and higher TSV failures.

As can be observed in Figure 5, RLARA is more stable and has smaller fluctuations, while TAFT is very unstable when used in varying fault models. The reason for this is that RLARA is TSV-aware and prefers to select the lowest TSV utilization path. The TSV usage can be reduced during the training process. However, the TSV usage in TAFT is fixed to the topology of 3D NoC with given faults. Meanwhile, the number of used vertical TSV links has a great deal to do with the variations in physical topology. It can cause back and forth if the topology is changed. Such a difference can be explained by the number of average hops shown in Table 5.

Table 5. TSV usage comparison for different routing algorithms using average hops.

Average Hops	Ideal	TAFT	RLARA	Improvement
$3 \times 3 \times 3$ NoC (5% faults)	2	2.8	2.73	3.41%
$4 \times 4 \times 4$ NoC (5% faults)	3	3.92	3.89	1.10%
$3 \times 3 \times 3$ NoC (15% faults)	2	2.9	2.80	5.11%
$4 \times 4 \times 4$ NoC (15% faults)	3	4.18	4.17	0.50%

The average number of RLARA in the overall average vertical links usage decreases in the range of about 0.5% to 5.11%. Therefore, RLARA is more suitable for fault-tolerant 3D NoC and is able to make a good trade-off between performance and reliability.

4.2.3. Adaptiveness under Fast Faults Changing

The successful delivery rate, the average latency and the training time under varying fault models are discussed to evaluate the adaptiveness of routing algorithms in the subsection.

On one hand, the good adaptiveness and fast recovery are reflected in the successful delivery rate under varying fault models. Figure 6 shows a comparison of the successful delivery rates of packets with increasing training rounds in $3 \times 3 \times 3$ and $4 \times 4 \times 4$ NoCs. The fault model with different fault links is changed every 70 training rounds, and the proportion of faults increases after the fault model is changed twice. Obviously, the successful packet delivery rate of RLARA increases significantly at the beginning and reaches a stable high point faster than TAFT. The successful delivery rate will not be much lower than 80% even after the fault model is changed. RLARA can recover and enable high-quality communication as soon as the training is continued.

On the other hand, the good adaptiveness and fast recovery are reflected in the average latency. Figure 7 shows a comparison of average latency between different routing algorithms when the fault model changes under different 3D NoC topologies. The lower the latency, the higher the performance. The ideal case provides the best routing path, and our proposed RLARA is between the ideal case and the worse TAFT. Some peaks in the results for RLARA are caused by the update of the routing selection when the topology was changed with a different fault model configuration. Changing the fault model twice represents how the average latency changes when the fault model changes. The Garnet simulator cannot change the topology of 3D NoC during one simulation. The red vertical line indicates that the running fault model has changed. At the very beginning, the low latency results are related to the initial training process, as is explained in Section 4.2.1. Each time the fault model is changed, the latency will be increased suddenly and then slowly drop, because each simulation includes some training rounds to ensure the dynamic characteristics. Therefore, the robust RLARA shows a lower average latency (1.06~5.63%) than TAFT under varying network sizes and fault models.

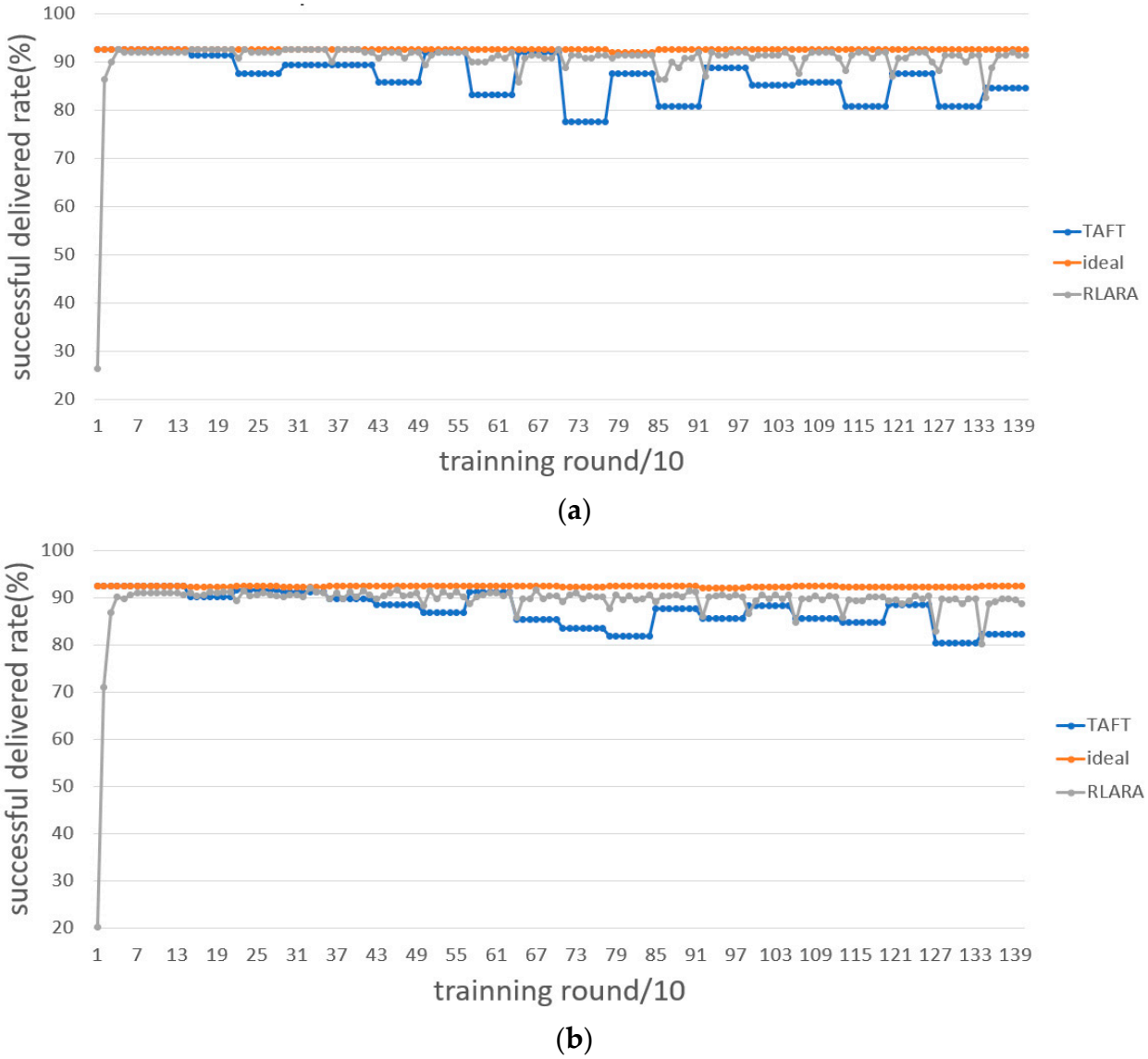
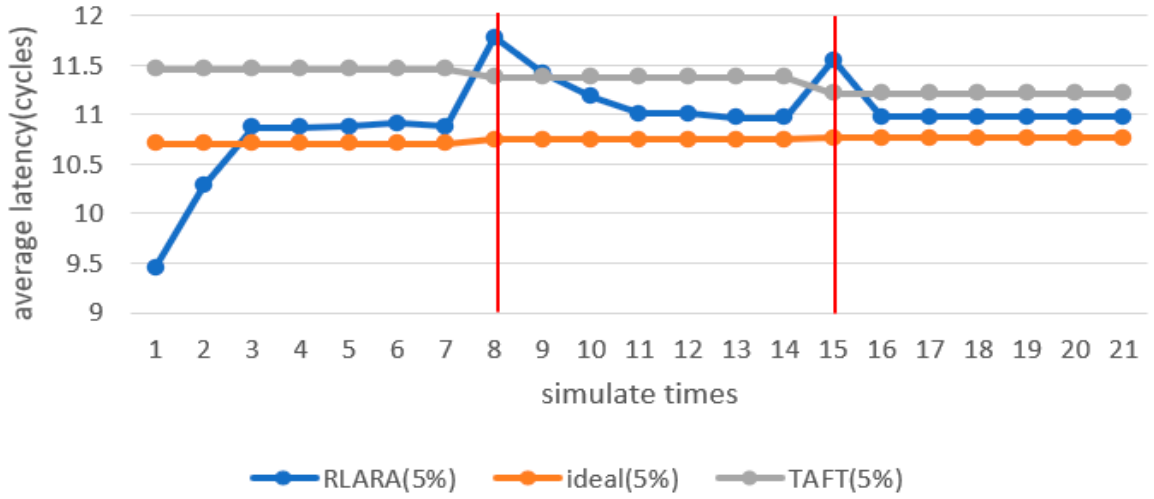
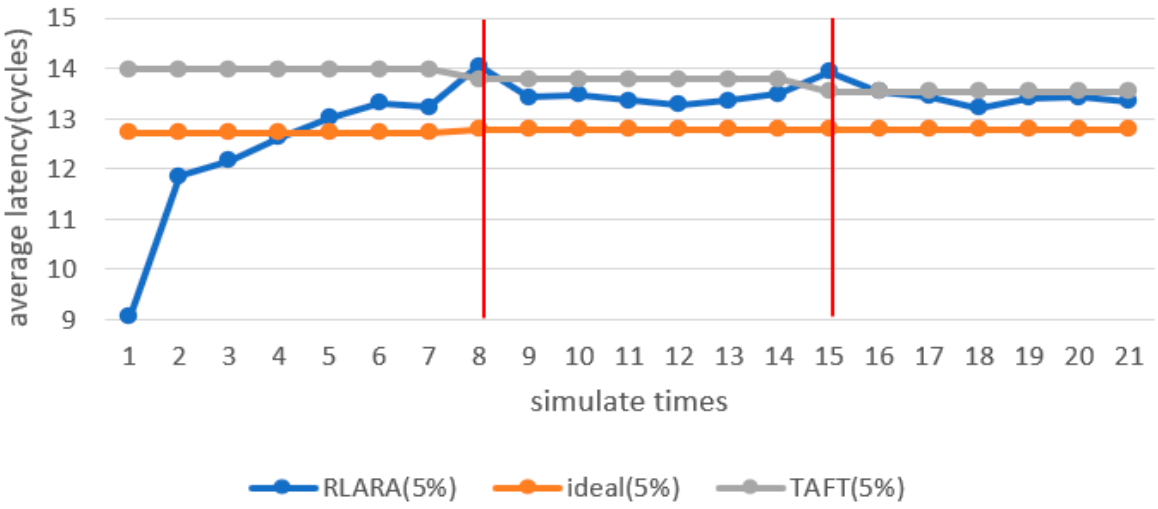


Figure 6. Packets successfully delivered rate comparison under varying fault models for different network sizes: (**a**) 3 × 3 × 3 NoC; (**b**) 4 × 4 × 4 NoC. The changing fault models give rise to different fault links and new topologies such that the routing algorithms need some time to recalculate the new routing paths. RLARA can provide the new routing decisions using both local and global network information.

The training time also influences the dynamic adaptiveness and speed of recovery. Figure 8 shows the training time variation under different topologies. It can be observed that the training time spent on 10 training rounds in a 3 × 3 × 3 3D NoC is very close to 0. Even in a 3D NoC with a scale of 4 × 4 × 4, the training time of 10 training rounds is less than 1 second. Each training round costs about 8 ms in 4 × 4 × 4 3D NoC, and even less than 1 ms in 3 × 3 × 3 3D NoC. This time gap results from the larger Q-Table of 4 × 4 × 4 than 3 × 3 × 3. Therefore, even though it is acceptable to periodically detect fault models and retrain them based on the original Q-Table, K-means clustering can further compress the Q-Table in order to save space and speed up recovery.



(**a**)



(**b**)

Figure 7. Average latency comparison under varying fault models for different network sizes: (**a**) 5% faults in 3 × 3 × 3 NoC; (**b**) 5% faults in 4 × 4 × 4 NoC. The lower the average latency, the better the performance. Compared with TAFT, the proposed RLARA can achieve lower average latency and approach to the ideal case. The peaks represent retraining for new routing decisions under a new fault model. The red vertical line indicates that the running fault model has changed.

The training time also influences the dynamic adaptiveness and speed of recovery. Figure 8 shows the training time variation under different topologies. It can be observed that the training time spent on 10 training rounds in a 3 × 3 × 3 3D NoC is very close to 0. Even in a 3D NoC with a scale of 4 × 4 × 4, the training time of 10 training rounds is less than 1 second. Each training round costs about 8 ms in 4 × 4 × 4 3D NoC, and even less than 1 ms in 3 × 3 × 3 3D NoC. This time gap results from the larger Q-Table of 4 × 4 × 4 than 3 × 3 × 3. Therefore, even though it is acceptable to periodically detect fault models and retrain them based on the original Q-Table, K-means clustering can further compress the Q-Table in order to save space and speed up recovery.

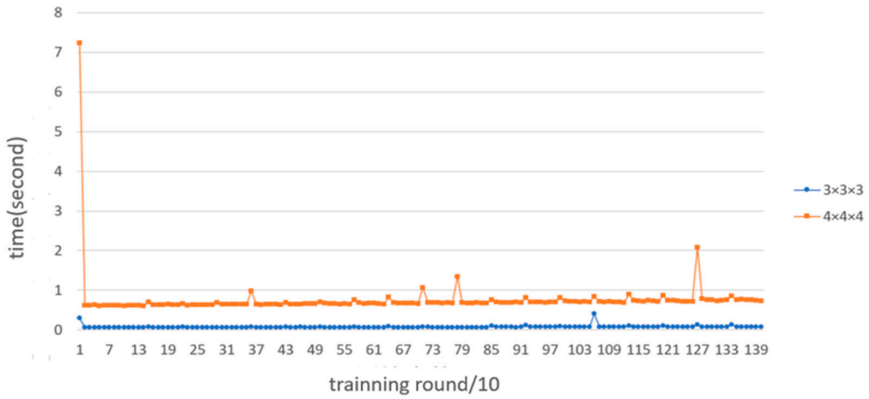
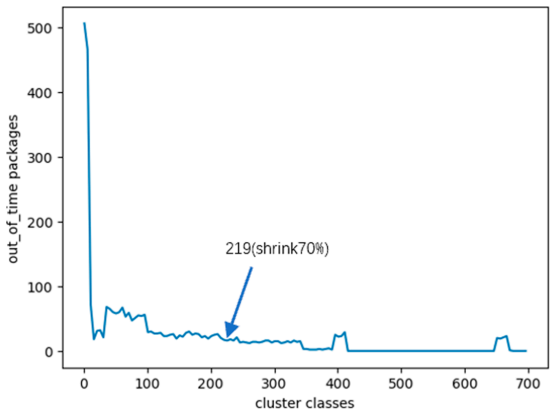


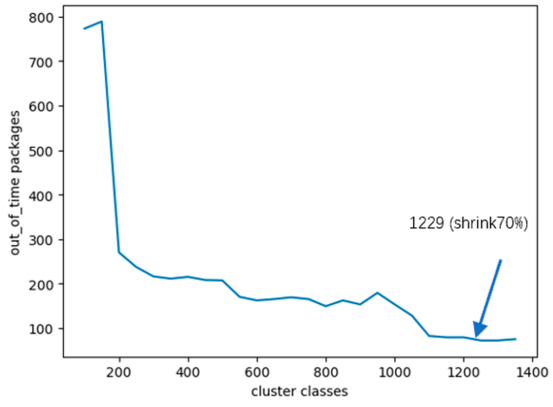
Figure 8. Short training time of RLARA for varying network sizes. The smaller the network size, the shorter the training time. Moreover, the early stage of training, devoted to selecting the routing path, requires much more time, while the following retraining is very fast and requires little recovery time for a new fault model.

4.3. K-Means Impacts on Q-Table

RLARA uses K-means clustering to reduce the size of the large Q-Table via exploiting the similarity of routing information. The K value has significant impacts on the routing results, as Figure 9 shows.



(**a**)

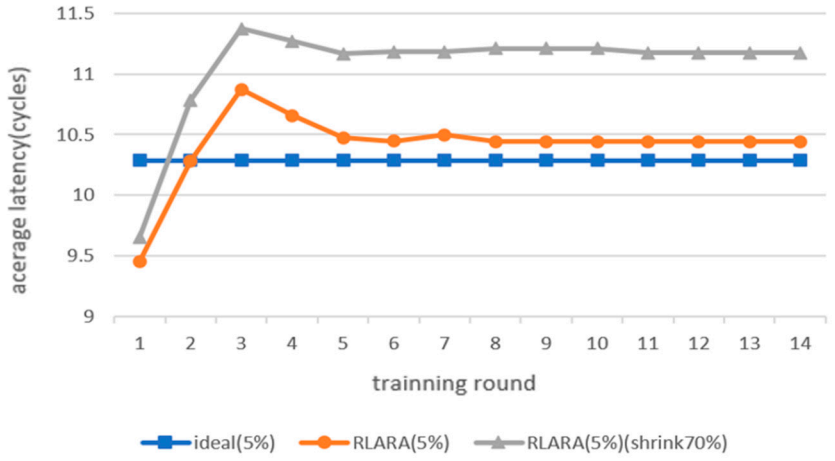


(**b**)

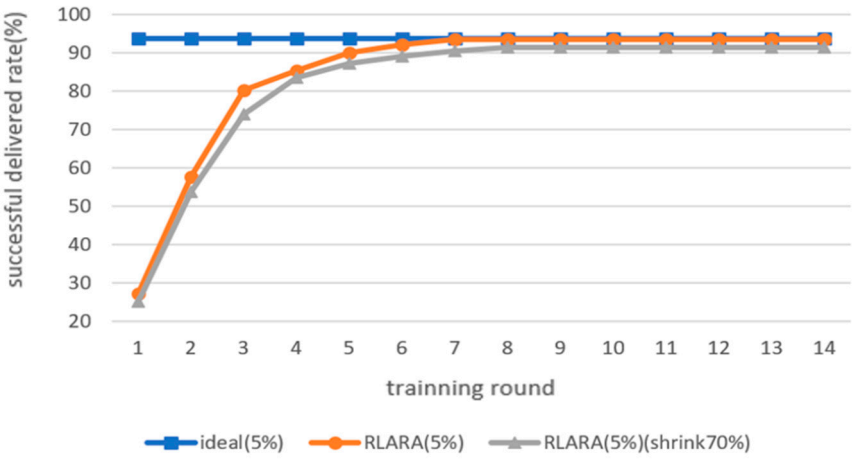
Figure 9. K-means impacts on packet timeouts under varying clustering groups: (**a**) 3 × 3 × 3 NoC; (**b**) 4 × 4 × 4 NoC. It is noted that a 70% compression rate can result in the lowest number of packet transmissions out-of-time; 219 and 1229 are the best K-values for the 3 × 3 × 3 and 4 × 4 × 4 networks, respectively.

Different optimal K values are selected for different network sizes of 3 × 3 × 3 (in Figure 9a) and 4 × 4 × 4 Mesh in (Figure 9b). The performance can be effectively guaranteed, even when the Q-Table is reduced by 70%, such that the number of packet timeouts is stable in a low range. Therefore, we consider 70% reduction as a basic RLARA and compare its performance with the original Q-Table.

As is shown in Figure 10, the average latency and successfully delivered rate are compared at a 5% fault injection rate between the Q-Table with the 70% reduction of RLARA and the Q-Table without compression. The degradation of both average latency and successful delivery rate occurs, but can be ignored after the Q-Table is compressed by K-means clustering in large-scale 3D NoCs.



(**a**)



(**b**)

Figure 10. Comparison of delay and successful delivery rate of Q-Tables without shrinking and after the reduction in the 3 × 3 × 3 NoC topology. (**a**) Average latency comparison with and without K-means clustering. (**b**) Successful delivery rate comparison with and without K-means clustering. The performance loss or gap between before and after the K-means compression of the Q-Table is nearly negligible because of the average latency and successful delivery rate.

Tables 6 and 7 show the performance gap between the original RALAR and the K-means-enhanced RALAR under varying fault injection rates, compression rates and network sizes. It can be noted that using the K-means-assisted Q-Table with 70% reduction, the performance gaps between both average latency and successful delivery rate range from 1.25% to 8.78% under varying fault injection rates, as shown in Table 6.

Table 6. Performance gap of the basic RALRA and the K-means-enhanced RALAR with a 70% compression rate and varying fault injection rates.

Fault Injection Rate		5%	10%	15%	20%
3 × 3 × 3 3DNoC	Average latency	7.74%	8.78%	7.69%	7.29%
	Successful delivery rate	1.25%	2.18%	3.49%	5.10%
4 × 4 × 4 3DNoC	Average latency	6.36%	7.11%	6.64%	7.40%
	Successful delivery rate	1.57%	3.78%	5.96%	6.30%

Table 7. Performance gap of the basic RALRA and the K-means-enhanced RALAR with a 5% fault injection rate and varying compression rates.

Compression Rate		30%	50%	70%	90%
3 × 3 × 3 3DNoC	Average latency	2.94%	6.71%	7.74%	19.52%
	Successful delivery rate	0.77%	0.56%	1.25%	27.29%
4 × 4 × 4 3DNoC	Average latency	2.42%	5.51%	6.36%	16.04%
	Successful delivery rate	0.54%	1.27%	1.57%	27.68%

Table 7 shows that the higher compression rates of 30%, 50% and 70% induce greater performance degradation, while a 90% compression rate decreases the performance metrics, with a 27.68% lower successful delivery rate and 19.52% higher average latency. The compression rate is critical to reducing the Q-Table in RLARA. Therefore, the trade-off between Q-Table compression and performance degradation is determined by a good K value. In a word, the K-means clustering method of RLARA is useful for reducing the size of the Q-Table on demand.

4.4. Deadlock Mitigation Analysis

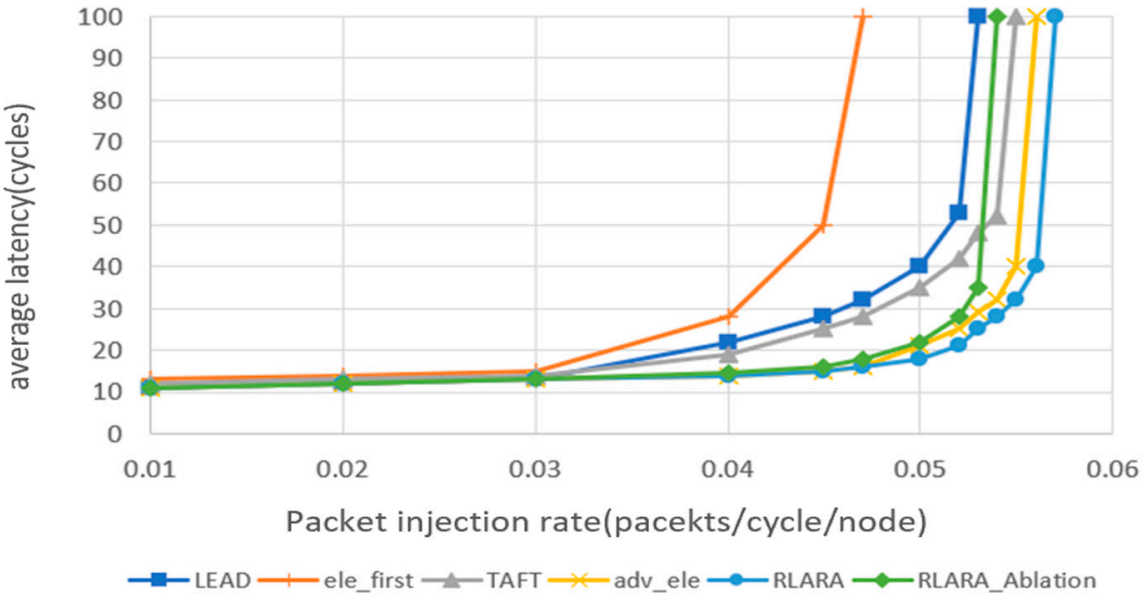
The deadlock issue often causes congestion and causes latency to increase dramatically. Therefore, we compare the proposed RLARA with various 3D NoC routing algorithms under varying traffic loads in Figure 11. The higher the packet injection rate, the higher the traffic load. Similarly to pre-existent works on LEAD [11] and Advertiser Elevator denoted as adv_ele [12], TAFT [16], and Elevator first denoted as ele_first [27], which were designed for partially connected TSVs, the faults here are only injected in TSVs. RLARA_Ablation a RLARA with no load balance scheme via the mixed status of the link and router.

The traffic load is measured by the packet injection rate. In this paper, a packet/node/cycle above 0.05 indicates high traffic load, and causes the latency to increase rapidly, as shown in Figure 11. Figure 11a,b give average latency comparisons for the 3 × 3 × 3 Mesh and 4 × 4 × 4 Mesh, respectively, and these show similar results. Firstly, it is to be expected that all the routing algorithms have low and stable latency under a low packet injection rate. However, the high packet injection rate causes packet blocking with large latency. More importantly, RLARA enters the saturation very late, and its latency is the lowest, compared with other routing algorithms. This ablation study shows that RLARA_Ablation performs better than LEAD and ele_first, but worse than TAFT, adv_ele and RLARA. Therefore, the mixed status load balance scheme assists RLARA in achieving good communication quality and mitigating the deadlock issue even under a high traffic load.

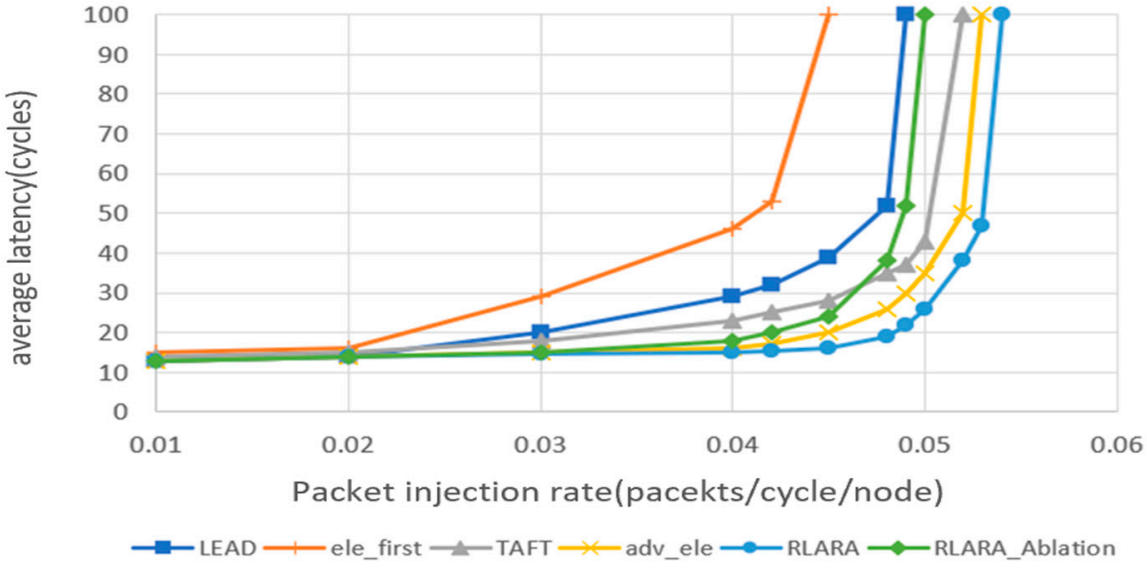
4.5. Discussion

Based on the above results and analysis, it is concluded that the proposed RLARA takes advantage of reinforcement learning to update the TSV-aware routing selection, and it performs better than LEAD, ele_first, TAFT and adv_ele. The lower average latency and higher successful delivery rates of RLARA under diverse configurations demonstrate its benefits when used in fault-tolerant 3D NoC architectures. Moreover, the issues of a large Q-Table and deadlock are also effectively solved. On one hand, the K-means clustering algorithm is adopted for compressing the Q-Table and minimizing the router's area

consumption. The 70% compression rate represents a good choice to balance the performance loss and area consumption. On the other hand, deadlock alleviation is achieved by using the mixed status of links and routers such that the proposed RLARA can achieve higher performance.



(**a**)



(**b**)

Figure 11. Average latency comparison of various routing algorithms under varying packet injection rates for 3D NoC: (**a**) 3 × 3 × 3 3D NoC; (**b**) 4 × 4 × 4 3D NoC. The deadlock mitigation of the proposed RLARA causes it to enter the saturation region later and achieve the lowest average latency compared to other routing algorithms.

It is also noted that this proposed approach uses a random fault injection and network pattern. More realistic applications, such as VOPD, MWD and JPEG, should be mapped into 3D NoC for further evaluation. On the other hand, the power and area consumptions

of the router in RLARA have not been estimated and optimized. We will focus on these two directions in future works.

5. Conclusions

A TSV-aware routing algorithm, RLARA, is here proposed for use in fault-tolerant 3D NoCs using the K-means enhanced reinforcement learning method globally and the mixed status load balance locally. The high failure rate of TSV vertical links is comprehensively considered in the routing decisions, while the various fault models are dynamically characterized via a reinforcement learning approach. The K-means clustering method can be used to further exploit the routing information similarity and reduce the Q-Table by 70%, with acceptable performance loss. The mixed status of links and routers is proven to mitigate the deadlock blocking issue well. The experimental results demonstrate that the proposed approach can achieve an average latency reduction of 9.04% and an improvement in successful delivery rate of 8.58% compared to the state-of-the-art for 3D NoCs. Additionally, TSV usage can be reduced by 5.11% to alleviate TSV overheating, while a strong communication performance can be achieved due to its good adaptiveness, with less than 1 s training time, even with the frequent changing of fault models. Therefore, the proposed RLARA represents an efficient fault-tolerant routing algorithm for use in 3D NoCs.

Author Contributions: Conceptualization, J.J. and R.S.; methodology, J.J.; software, L.C.; validation, J.J., R.S. and L.C.; formal analysis, R.S..; investigation, R.S.; resources, J.J.; data curation, L.C.; writing—original draft preparation, J.J. and L.C.; writing—review and editing, J.J. and R.S.; visualization, R.S.; supervision, D.H.; project administration, D.H. and J.L.; funding acquisition, Jiajia Jia and J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Shanghai Pujiang Talent Program (No. 21PJD026), National Key Research and Development Program of China (No. 2021YFC2801000), and the Major Research plan of the National Social Science Foundation of China (No. 20&ZD130).

Data Availability Statement: Data are contained within the article.

Acknowledgments: Thanks to the editor and anonymous reviewers for giving constructive suggestions and helping to improve the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Topol, A.W.; La Tulipe, D.C.; Shi, L.; Frank, D.J.; Bernstein, K.; Steen, S.E.; Kumar, A.; Singco, G.U.; Young, A.M.; Guarini, K.W.; et al. Three-dimensional integrated circuits. *IBM J. Res. Dev.* **2006**, *50*, 491–506. [CrossRef]
2. Liu, C.; Zhang, L.; Han, Y.; Li, X. Vertical interconnects squeezing in symmetric 3D mesh Network-on-Chip. In Proceedings of the 16th Asia and South Pacific Design Automation Conference, Yokohama, Japan, 25–28 January 2011; pp. 357–362.
3. Feero, B.S.; Pande, P.P. Networks-on-chip in a three-dimensional environment: A performance evaluation. *IEEE Trans. Comput.* **2008**, *58*, 32–45. [CrossRef]
4. Syal, N.; Sehgal, V.K. Qualitative analysis of 3D routing algorithms in 3 × 3 × 3 mesh NoC topology under varying load in Bio-SoC. *Int. J. E-Health Med. Commun.* **2020**, *11*, 86–102. [CrossRef]
5. Khayambashi, M.; Yaghini, P.M.; Eghbal, A.; Bagherzadeh, N. Analytical reliability analysis of 3D NoC under TSV failure. *ACM J. Emerg. Technol. Comput. Syst.* **2015**, *11*, 1–16. [CrossRef]
6. Eghbal, A.; Yaghini, P.M.; Bagherzadeh, N.; Khayambashi, M. Analytical fault tolerance assessment and metrics for TSV-based 3D network-on-chip. *IEEE Trans. Comput.* **2015**, *64*, 3591–3604. [CrossRef]
7. Jheng, K.Y.; Chao, C.H.; Wang, H.Y.; Wu, A.Y. Traffic-thermal mutual-coupling co-simulation platform for three-dimensional network-on-chip. In Proceedings of the 2010 International Symposium on VLSI Design, Automation and Test, Hsinchu, Taiwan, 26–29 April 2010; pp. 135–138.
8. Taouil, M.; Hamdioui, S.; Verbree, J.; Marinissen, E.J. On maximizing the compound yield for 3D wafer-to-wafer stacked ICs. In Proceedings of the 2010 IEEE International Test Conference, Austin, TX, USA, 2–4 November 2010; pp. 1–10.
9. Pasricha, S. Exploring serial vertical interconnects for 3D ICs. In Proceedings of the 46th Annual Design Automation Conference, San Francisco, CA, USA, 26–31 July 2009; pp. 581–586.
10. Davis, W.; Wilson, J.; Mick, S.; Xu, J.; Hua, H.; Mineo, C.; Sule, A.; Steer, M.; Franzon, P. Demystifying 3D ICs: The pros and cons of going vertical. *IEEE Des. Test Comput.* **2005**, *22*, 498–510. [CrossRef]

11. Salamat, R.; Khayambashi, M.; Ebrahimi, M.; Bagherzadeh, N. LEAD: An adaptive 3D-NoC routing algorithm with queuing-theory based analytical verification. *IEEE Trans. Comput.* **2018**, *67*, 1153–1166. [CrossRef]
12. Taheri, E.; Isakov, M.; Patooghy, A.; Kinsy, M.A. Advertiser elevator: A fault tolerant routing algorithm for partially connected 3D Network-on-Chips. In Proceedings of the 2017 IEEE 60th International Midwest Symposium on Circuits and Systems, Boston, MA, USA, 6–9 August 2017; pp. 136–139.
13. Da Silva, A.A.; Junior, L.M.E.S.; Coelho, A.; Silveira, J.; Marcon, C. Reflect3d: An Adaptive and Fault-Tolerant Routing Algorithm for Vertically-Partially-Connected 3D-NoC. In Proceedings of the 2021 34th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design, Campinas, Brazil, 23–27 August 2021; pp. 1–6.
14. Nezarat, M.; Momeni, M. TCAR: Thermal and Congestion-Aware Routing Algorithm in a Partially Connected 3D Network on Chip. In Proceedings of the 2022 12th International Conference on Computer and Knowledge Engineering, Mashhad, Iran, 17–18 November 2022; pp. 106–111.
15. Taheri, E.; Kim, R.G.; Nikdast, M. AdEle+: An Adaptive Congestion-and-Energy-Aware Elevator Selection for Partially Connected 3D NoCs. *IEEE Trans. Comput.* **2023**, *72*, 2278–2292. [CrossRef]
16. Meyer, M.C.; Wang, Y.; Watanabe, T. Fault-tolerant traffic-aware routing algorithm for 3-D photonic networks-on-chip. In Proceedings of the 2019 IEEE 13th International Symposium on Embedded Multicore/Many-core Systems-on-Chip, Singapore, 1–4 October 2019; pp. 172–179.
17. Li, J.; Qin, C.; Sun, X. An efficient adaptive routing algorithm for the Co-optimization of fault tolerance and congestion awareness based on 3D NoC. *Microelectron. J.* **2023**, *142*, 105989. [CrossRef]
18. Ahmed, A.B.; Abdallah, A.B. Graceful deadlock-free fault-tolerant routing algorithm for 3D Network-on-Chip architectures. *J. Parallel Distrib. Comput.* **2014**, *74*, 2229–2240. [CrossRef]
19. Ebrahimi, M.; Daneshtalab, M.; Plosila, J. Fault-tolerant routing algorithm for 3D NoC using hamiltonian path strategy. In Proceedings of the 2013 Design, Automation & Test in Europe Conference & Exhibition, Grenoble, France, 18–22 March 2013; pp. 1601–1604.
20. Bölücü, N.; Tosun, S. Q-Learning-based Routing Algorithm for 3D Network-on-Chips. In Proceedings of the 2021 24th International Symposium on Design and Diagnostics of Electronic Circuits & Systems, Vienna, Austria, 7–9 April 2021; pp. 33–36.
21. Shahabinejad, N.; Beitollahi, H. Q-thermal: A Q-learning-based thermal-aware routing algorithm for 3-D network on-chips. *IEEE Trans. Compon. Packag. Manuf. Technol.* **2020**, *10*, 1482–1490. [CrossRef]
22. Ahmed, A.B.; Abdallah, A.B. LA-XYZ: Low latency, high throughput look-ahead routing algorithm for 3D network-on-chip (3D-NoC) architecture. In Proceedings of the 2012 IEEE 6th International Symposium on Embedded Multicore SoCs, Aizu-Wakamatsu, Japan, 20–22 September 2012; pp. 167–174.
23. Ben, R.; Ge, F.; Tong, X.; Wu, N.; Zhang, Y.; Zhou, F. A multicast routing algorithm for 3D network-on-chip in chip multi-processors. In Proceedings of the World Congress on Engineering, London, UK, 4–6 July 2016; Volume 1.
24. Ebrahimi, M.; Chang, X.; Daneshtalab, M.; Plosila, J.; Liljeberg, P.; Tenhunen, H. DyXYZ: Fully adaptive routing algorithm for 3D NoCs. In Proceedings of the 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, Belfast, UK, 27 February–1 March 2013; pp. 499–503.
25. Samala, J.; Takawale, H.; Chokhani, Y.; Veda Bhanu, P.; Soumya, J. Fault-Tolerant Routing Algorithm for Mesh based NoC using Reinforcement Learning. In Proceedings of the 2020 International Symposium on VLSI Design and Test (VDAT), Bhubaneswar, India, 23–25 July 2020; pp. 1–6.
26. Jagadheesh, S.; Veda Bhanu, P.; Soumya, J.; Cenkeramaddi, L.R. Reinforcement Learning Based Fault-Tolerant Routing Algorithm for Mesh Based NoC and Its FPGA Implementation. *IEEE Access* **2022**, *10*, 44724–44737. [CrossRef]
27. Dubois, F.; Sheibanyrad, A.; Petrot, F.; Bahmani, M. Elevator-first: A deadlock-free distributed routing algorithm for vertically partially connected 3D-NOCS. *IEEE Trans. Comput.* **2013**, *62*, 609–615. [CrossRef]
28. Pasricha, S.; Zou, Y. A low overhead fault tolerant routing scheme for 3D Networks-on-Chip. In Proceedings of the 12th International Symposium on Quality Electronic Design, Santa Clara, CA, USA, 14–16 March 2011; pp. 1–8.
29. Sinha, D.; Roy, A.; Kumar, K.V.; Kulkarni, P.; Soumya, J. Dn-FTR: Fault-tolerant routing algorithm for Mesh based network-on-chip. In Proceedings of the 4th International Conference on Recent Advances in Information Technology, Dhanbad, India, 15–17 March 2018; pp. 1–5.
30. Jiang, X.; Zeng, L.; Watanabe, T. A sophisticated routing algorithm in 3D Noc with fixed TSVS for low energy and latency. *Inf. Media Technol.* **2014**, *9*, 404–412. [CrossRef]
31. Ying, H.; Jaiswal, A.; Hofmann, K. Deadlock-free routing algorithms for 3-dimension networks-on-chip with reduced vertical channel density topologies. In Proceedings of the 2012 International Conference on High Performance Computing & Simulation, Madrid, Spain, 2–6 July 2012; pp. 268–274.
32. Vu, T.H.; Okuyama, Y.; Abdallah, A.B. Comprehensive analytic performance assessment and K-means based multicast routing algorithm and architecture for 3D-NoC of spiking neurons. *ACM J. Emerg. Technol. Comput. Syst.* **2019**, *15*, 1–28. [CrossRef]
33. Chen, L.; Jiao, J.; Shen, R. KARL: A Cost-effective Routing Algorithm in Fault Tolerant 3D Network-on-Chip via K-means Assisted Reinforcement Learning. In Proceedings of the 2022 7th International Conference on Big Data and Computing, Shenzhen, India, 27–29 May 2022; pp. 45–50.
34. Fang, J.; Cai, H.; Lv, X. Hybrid Optimization Algorithm Based on Double Particle Swarm in 3D NoC Mapping. *Micromachines* **2023**, *14*, 628. [CrossRef] [PubMed]

35. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2018.
36. Reza, M.F.; Le, T.T. Reinforcement Learning Enabled Routing for High-Performance Networks-on-Chip. In Proceedings of the 2021 IEEE International Symposium on Circuits and Systems (ISCAS), Daegu, Republic of Korea, 22–28 May 2021; pp. 1–5.
37. Agarwal, N.; Krishna, T.; Peh, L.S.; Jha, N.K. GARNET: A detailed on-chip network model inside a full-system simulator. In Proceedings of the 2009 IEEE International Symposium on Performance Analysis of Systems and Software, Boston, MA, USA, 26–28 April 2009; pp. 33–42.
38. Kao, S.-C.; Yang, C.-H.H.; Chen, P.-Y.; Ma, X.; Krishna, T. Reinforcement learning based interconnection routing for adaptive traffic optimization. In Proceedings of the 13th IEEE/ACM International Symposium on Networks-on-Chip (NOCS '19), New York, NY, USA, 17–18 October 2019; pp. 1–2.