




Article

A Machine Learning Mapping Algorithm for NoC Optimization

Xiaodong Weng ^{1,*} , Yi Liu ¹, Changqing Xu ² , Xiaoling Lin ³, Linjun Zhan ², Shun Yao Wang ², Dongdong Chen ¹  and Yintang Yang ¹

¹ School of Microelectronics, Xidian University, Xi'an 710126, China

² Guangzhou Institute of Technology, Xidian University, Guangzhou 510555, China

³ Technology on Reliability Physics and Application Technology of Electronic Component Laboratory, China Electronic Product Reliability and Environmental Testing Research Institute, Guangzhou 510610, China

* Correspondence: xdweng@stu.xidian.edu.cn

Abstract: Network on chip (NoC) is a promising solution to the challenge of multi-core System-on-Chip (SoC) communication design. Application mapping is the first and most important step in the NoC synthesis flow, which determines most of the NoC design performance. NoC mapping has been confirmed as an NP-hard (Non-Polynomial hard) problem, which could not be solved in polynomial time. Various heuristic mapping algorithms have been applied to the mapping problem. However, the heuristic algorithm easily falls into a local optimal solution which causes performance loss. Additionally, regular topologies of NoC, such as the ring, torus, etc., may generate symmetric solutions in the NoC mapping process, which increase the performance loss. Machine learning involves data-driven methods to analyze trends, find relationships, and develop models to predict things based on datasets. In this paper, an NoC machine learning mapping algorithm is proposed to solve a mapping problem. A Low-complexity and no symmetry NoC mapping dataset is defined, and a data augmentation approach is proposed to build dataset. With the dataset defined, a multi-label machine learning is established. The simulation results have confirmed that the machine learning mapping algorithm is proposed have at least 99.6% model accuracy and an average of 96.3% mapping accuracy.

Keywords: network on chip; application mapping; heuristic algorithm; machine learning mapping algorithm



Citation: Weng, X.; Liu, Y.; Xu, C.; Lin, X.; Zhan, L.; Wang, S.; Chen, D.; Yang, Y. A Machine Learning Mapping Algorithm for NoC Optimization. *Symmetry* **2023**, *15*, 593. <https://doi.org/10.3390/sym15030593>

Academic Editors: Mihai Postolache and Jan Awrejcewicz

Received: 13 January 2023

Revised: 17 February 2023

Accepted: 22 February 2023

Published: 25 February 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the increase in integrated circuits (ICs) power consumption in the past few decades, some ICs finally reached their fundamental thermal limit at the beginning of the past decade [1]. System on chip (SoC) has begun to shift from a high-performance single-core design into a multi-core design [2], which follows the communication challenge. Network on chip (NoC) is one of the mainstream solutions for multicore SoC communication design [3,4]. NoC is an on-chip communication network based on packet switching [5], which consists of Resource Network Interfaces (RNI), routers, and interconnecting links [6].

NoC brings more communication bandwidth and increases communication cost. In 100 nm technology, the percentage of power consumption caused by communication between IP (Intellectual Property) cores exceeds 30% of the total power consumption. This value increases with more advanced technology and higher die integration [3,4]. Communication delay and power consumption are two important aspects of NoC design [7].

Application mapping is the first and most important step in the NoC design flow [7–9]. A good mapping solution will lead to a low communication cost. NoC application mapping has proven to be an NP-hard problem. Consequently, no exact algorithm is expected to solve the problem in a polynomial time, and even small instances may require considerable computation time [7,10,11].

The heuristic method is a natural and useful method to provide high-quality local optimal solutions [12,13]. Various heuristic mapping algorithms have been proposed to solve the application mapping problem, such as NMAP (Near-optimal Mapping), PSO (Particle Swarm Optimization), SA (Simulated Annealing Algorithm), GA (Genetic Algorithm), and ACO (Ant Colony Optimization) [14–22]. However, heuristic mapping technologies with limited computation resources easily fall into a local optimal solution, which causes performance loss [23–25]. Additionally, the symmetry of NoC mapping solutions may influence the performance of the heuristic NoC mapping algorithms, which increase performance loss.

Machine learning involves data-driven methods to analyze trends, find relationships, and develop models to predict things based on datasets [24,25]. Various NoC machine learning technology examples of research have been reported [26–30]. Refs. [26,27] use machine learning technology to predict the performance of NoC mapping. Refs. [28–30] use machine learning to explore the design space of NoC. More and more examples of research have applied machine learning technology to network on chip.

In this paper, a NoC machine learning mapping algorithm is proposed to solve a mapping problem. The space complexity of the existing mapping solution description [5,26,31] is too large for a machine learning model to fit with few samples. A low-complexity NoC mapping dataset is defined, and a data augmentation approach is proposed. With the characteristics of the dataset defined, a multi-label machine learning model is established.

2. The Methods of Establishing Machine Learning Mapping Algorithm

In this section, a machine learning mapping algorithm is presented. Firstly, an NoC mapping performance model is established. Secondly, symmetry analysis and space complexity of mapping solution are presented. Thirdly, the process of mapping dataset construction is detailed. Finally, a multi-label machine learning model is established.

2.1. Problem Formulation

NoC mapping aims to assign NoC cores that minimize the energy consumption and communication delay. Referring to [15,32], the NoC mapping problem is formulated with the following definitions.

2.1.1. Characteristics Definition

“Definition 1: The Application Characteristic Graph (APCG). The application task is modeled as a directed graph $G(C, A)$, where each vertex $c_i \in C$ represents an IP core, each edge $a_{ij} \in A$ represents the communication between c_i and c_j , and the weight of each edge V_{ij} indicates the communication volume on edge a_{ij} [32]”. APCG is shown in Figure 1a.

“Definition 2: The Architecture Characterization Graph (ARCG). The NoC architecture is modeled as a directed graph $G(R, L)$, in which nodes of the graph represents routers $r_i \in R$ and the edges between nodes $l_{ij} \in L$ represents physical links between routers [32]”. ARCG is shown in Figure 1b.

“Definition 3: The Channel Communication Graph (CHCG). The NoC architecture is modeled as a directed graph $G(R, L)$, in which nodes of the graph represents routers $r_i \in R$ and the edges between nodes $L_{ij} \in L$ represents the channel load of the link after application mapping between router r_i and r_j which is the result calculated with the application mapping result and routing algorithm and constrained by the physical links l_{ij} and the bandwidth Bw_{ij} [32]”. CHCG is shown in Figure 1c.

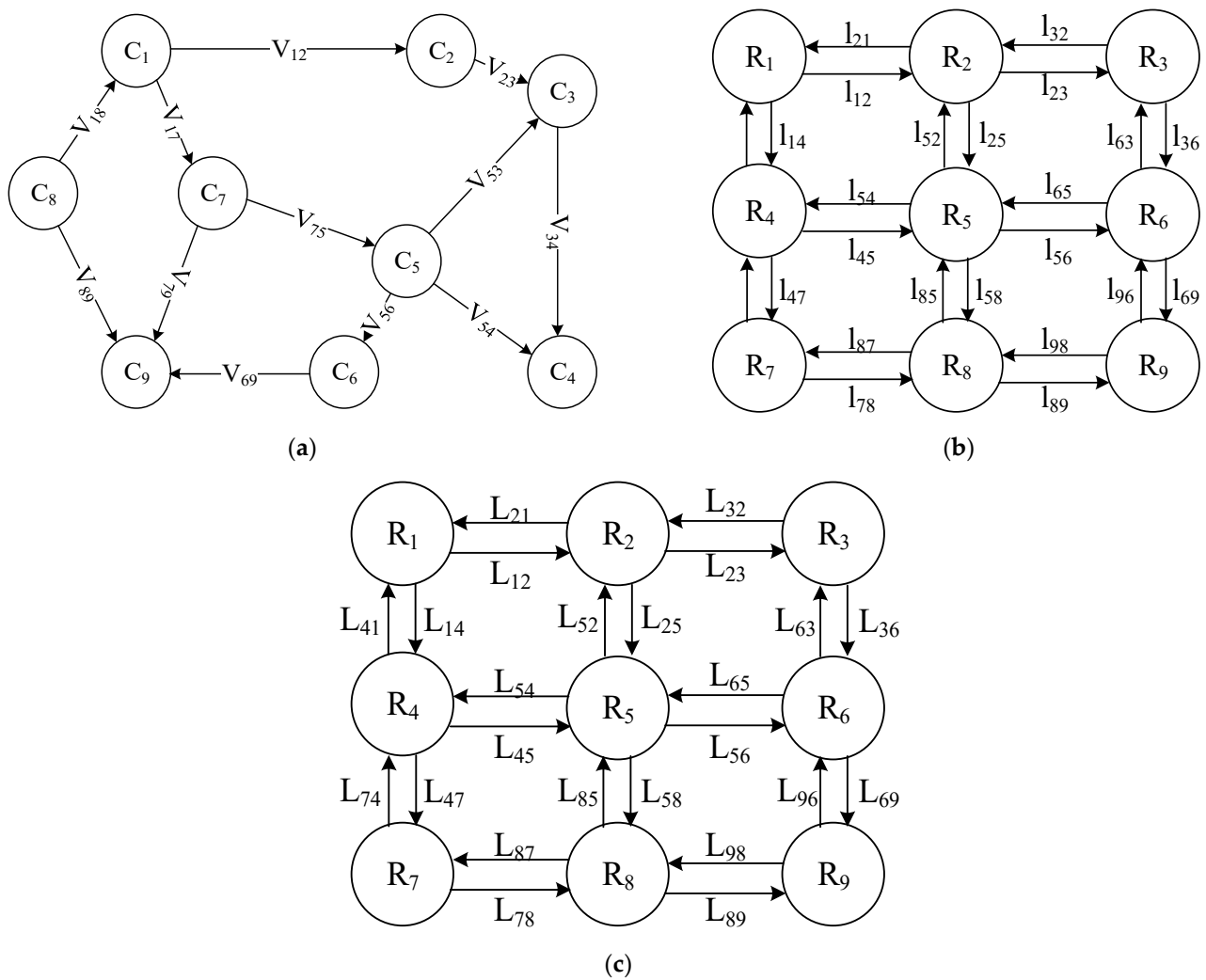


Figure 1. Characteristics: (a) APCG, (b) ARCG, and (c) CHCG [32].

2.1.2. Energy Model

In order to formulate the NoC energy consumption, one-bit energy [25] is introduced. r_i means i_{th} router. The energy of sending one-bit data from r_i to r_j is E_{ij} , which is formulated by (1).

$$E_{ij} = (\text{Hops} - 1) \times E_{lbit} + \text{Hops} \times E_{Rbit} \quad (1)$$

where E_{lbit} represents one-bit data energy consumption transmitted through a link, E_{Rbit} represents one-bit data energy consumption transmitted through a router, E_{Rbit} includes buffer and switch energy consumption. Hops is the Manhattan distance from the sending node (x_i, y_i) to the receiving node (x_j, y_j) and is given by (2).

$$\text{Hops} = |x_i - x_j| + |y_i - y_j| \quad (2)$$

Based on Equation (1), communication energy E_C is calculated by (3).

$$E_C = \sum b_{ij} \times E_{ij} \quad (3)$$

Here, b_{ij} is the count of bits sent from r_i to r_j . Referring to [5], energy consumption can be calculated with the bit energy values for link, switch, read and write buffer as 0.449 pJ, 0.284 pJ, 1.056 pJ, and 2.831 pJ, respectively.

2.1.3. Delay Model

The switching technology for the network is wormhole. The communication latency of NoC can be estimated by the average network delay (T_{av}) [27], which is introduced as (4).

$$T_{av} = \frac{T_l + T_r}{\sum_{i=1}^n \sum_{j=1}^n \lambda_{ij}} \quad (4)$$

where T_l and T_r are the delay from routers and links. Router transfers information by data packages. A data package is composed of several flits. λ_{ij} represents the flits quantity, which are sent from c_i to c_j . n represents the amount of Processing Element (PE). c_i means the i_{th} core in NoC topology.

At the transfer level, the latency of global links can be calculated by (5).

$$T_l = \sum_i^m \sum_j^n \lambda_{ij} C_{ij} u_t \quad (5)$$

where i and j are the node indexes of the sending node and receiving node. C_{ij} is the minimum count of links required, which are used to send packets from the sending node to receiving node, and u_t is a unit of time.

The latency of m-ports queuing router has been established in [29] as (6).

$$T_{router} = \frac{7 + T_{aq}}{2} u_t \quad (6)$$

where T_{aq} is the average count of time steps spent by a flit in the queue. T_{aq} has been modeled by [5], shown as (7).

$$T_{aq} = \frac{B_{av}}{N_0} \quad (7)$$

where B_{av} is the average size of queue, and N_0 is the throughput (packet/time step). Referring to [5,26], they can be calculated out. The total router delay is formulated as (8).

$$T_r = \sum_{i=1}^m \sum_{j=1}^n \left(\lambda_{ij} \sum_{x=1}^{hops} T_{router}(ports) \right) \quad (8)$$

2.1.4. Optimization Model

Energy and delay are two important performance optimization targets. In order to meet the requirements of different design, the performance cost function can be expressed as (9).

$$Cost = (\alpha NE_c + (1 - \alpha) NT_l) \quad (9)$$

where $\alpha \in [0, 1]$ is a weight parameter, and NE_c and NT_l are the normalized energy and delay, respectively.

2.2. Details of Machine Learning Mapping Algorithm

In this section, the space complexity and symmetry of NoC mapping solutions are analyzed. Based on the analysis, a low-complexity NoC mapping dataset with no symmetry is proposed. Considering the difficulty of searching global results of NoC mapping problems, a data augmentation approach is proposed to generated samples with a globally searched sample.

2.2.1. Space Complexity and Symmetry

On the existing examples of research, NoC mapping solutions are defined as a sequence. For example, Figure 2 is the VOPD (Video Object Plane Decoder) NoC mapping problem.

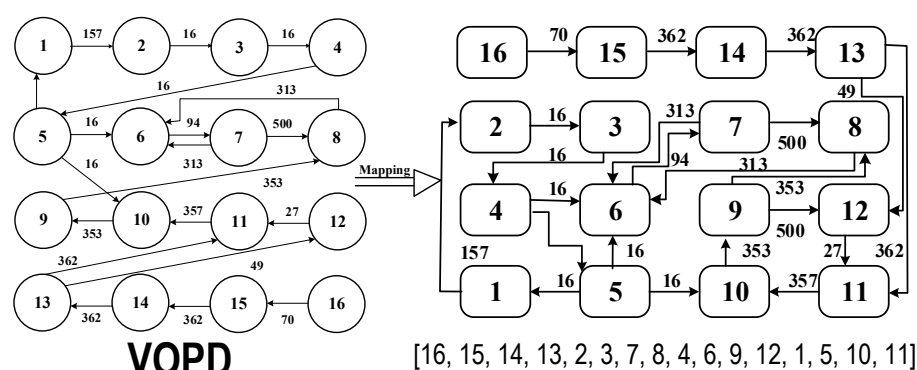


Figure 2. VOPD application mapping.

Application VOPD is mapping on a 4×4 mesh NoC. One of its solutions is described as [1, 5, 10, 11, 4, 6, 9, 12, 2, 3, 7, 8, 16, 15, 14, 13]. The space complexity of the VOPD mapping solution sequence is $16!$, which means it has $16!$ mapping solutions. Mapping algorithms are designed to search the solution space and find the optimal one. The space complexity of an n -core mapping solution sequence is $n!$, which is too large for a machine learning model to fit with a few-samples dataset.

Network topology with the same routers and links is symmetric, such as torus, ring, etc. Symmetry is a property of network topology. It means that the network will not change whether it is rotated or symmetrical. Topology symmetry will cause mapping solution symmetry. For example, Figure 3. is the VOPD NoC mapping problem. [1, 5, 10, 11, 4, 6, 9, 12, 2, 3, 7, 8, 16, 15, 14, 13] is one mapping sequence. [16, 15, 14, 13, 2, 3, 7, 8, 4, 6, 9, 12, 1, 5, 10, 11] is a symmetric sequence which can be obtained through the network turning upside down. Both mapping sequences have the same network information and performance. This characteristic is called as NoC mapping symmetry. The symmetry of NoC mapping could influence the machine learning approach construction, as shown in Figure 3.

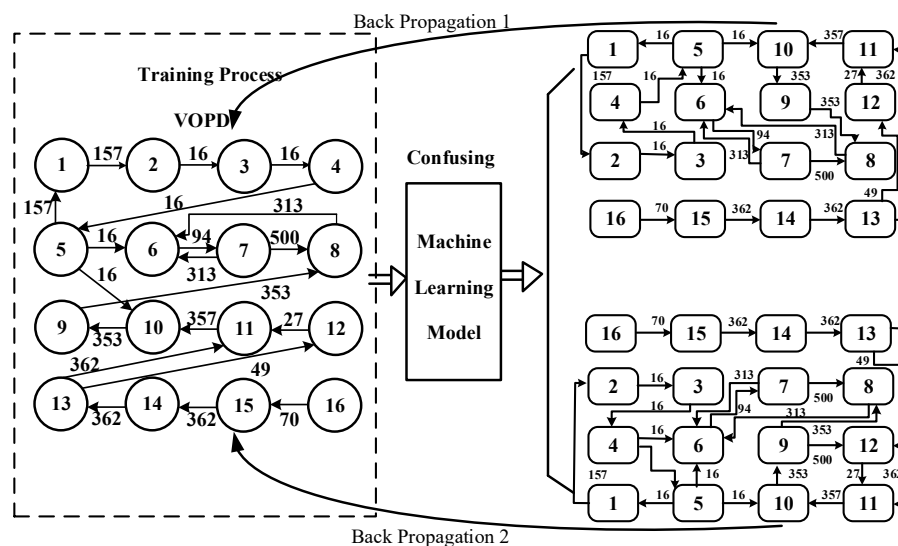


Figure 3. Sequence makes machine learning model confused.

From Figure 3, let us assume the sequence [16, 15, 14, 13, 2, 3, 7, 8, 4, 6, 9, 12, 1, 5, 10, 11] is the optimal VOPD NoC mapping solution. After flipping the solution up and down, sequence [1, 5, 10, 11, 4, 6, 9, 12, 2, 3, 7, 8, 16, 15, 14, 13] is the optimal solution as well. Two symmetrical solutions cause the machine learning model to be confused by labeled signs. In machine learning, multiple inputs can correspond to one output of the same kind, but one input cannot correspond to multiple outputs of the same kind.

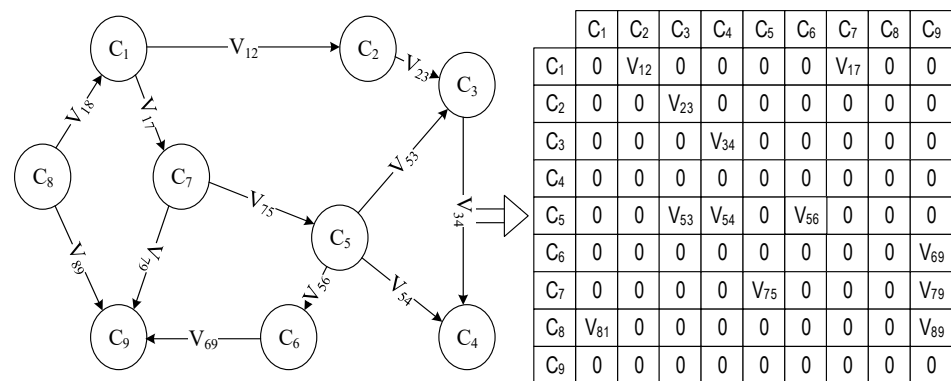
2.2.2. A Low-Complexity and No Symmetry NoC Mapping Dataset

NoC mapping is a problem to find appropriate application mapping solution on NoC topology. The inputs of this problem are application task graph and network topology information, and the output of this problem is the mapping sequence or corresponding mapping solution. NoC mapping performance could be calculated out with mapping sequence and network information.

The space complexity of mapping sequence is too large, which means that the problem needs a large machine learning dataset.

In order to solve the dataset space complexity and the symmetry of mapping sequence, the point-to-point task matrix is defined as the input data of the dataset and the column vector of the nearest neighbor matrix is defined as the output data of the dataset.

The Point-to-Point Task Matrix: The task graph is modeled as a matrix, where $c_i \in C$ represents intellectual property (IP) core i , and each element $V_{ij} \in V$ in the matrix represents the communication volume between c_i and c_j , as Figure 4 shows.



In the ring NoC, the space complexity is C_n^2 . In the torus topology, the space complexity is C_n^4 . The space complexity of the nearest neighbor matrix is much smaller than the space complexity of sequence.

2.2.3. A Data Augmentation Approach

The NoC mapping problem is an NP-hard problem, which means that optimal mapping solution is hard to obtain. It is too expensive to directly generate datasets that meet the requirement of machine learning.

A data augmentation approach is proposed to augment datasets with few globally searched samples. The approach is shown below (Algorithm 1):

Algorithm 1 Data Augmentation with Number Exchange

Input: task graph and optimal mapping solution sequence

Output: a group of task graphs and corresponding optimal mapping solution sequences

01: {B} is the optimal mapping solution sequence of task B, calculated with global search;

02: Fetch data in an integer space $[1, n]$ without repetition to build an n -dimensional order sequence. $n!$ sequences could be obtained.

03: Let the initial order sequence be $\{1, 2, 3, \dots, n\}$.

04: Loop begins, $i = 2$,

05: the i th order sequence is $\{i1, i2, i3, \dots, in\}$.

06: Loop begins, $j = 1$,

07: Swap the number B_i in task graph B and {B} with number B_{ij} .

08: If $j = n$, end Loop.

09: If $i = n!$, end Loop.

10: $n!$ task graphs and corresponding optimal mapping solution sequences have been obtained.

For example, the optimal solution of task graph G2 (C, A) can be obtained from G1 (C, A) with its optimal solution, as Figure 6 shows. By swapping the identifier of C_1 and C_5 , the task graph G2 (C, A) can be obtained from G1 (C, A). With the same changing, the optimal mapping solution of G2 can be obtained from G1. In this way, $n!$ task graphs with corresponding optimal solutions can be obtained from G1 (C, A) with its optimal solution.

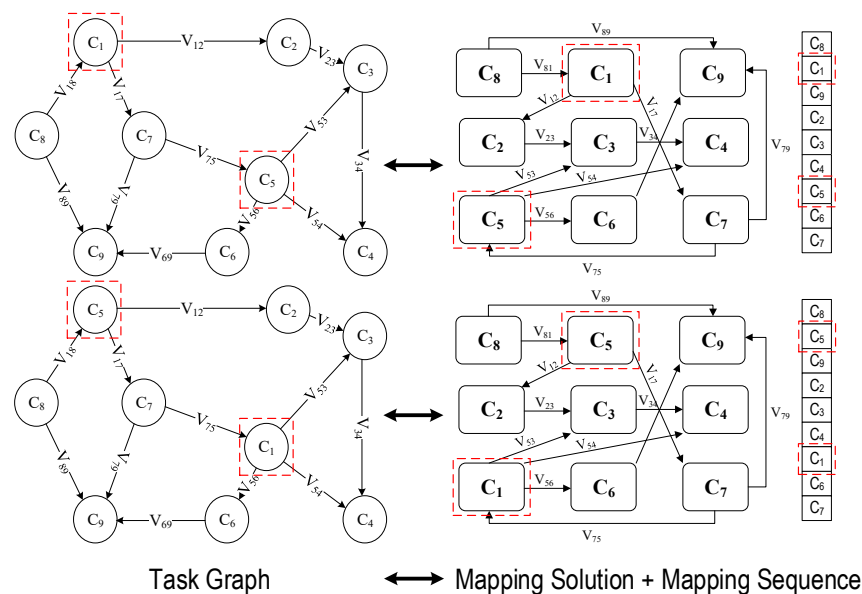


Figure 6. An example of data augmentation.

However, datasets established in this way have the risk of model fitting to the process of the data augmentation approach instead of fitting to the global search process, as Figure 7 shows.

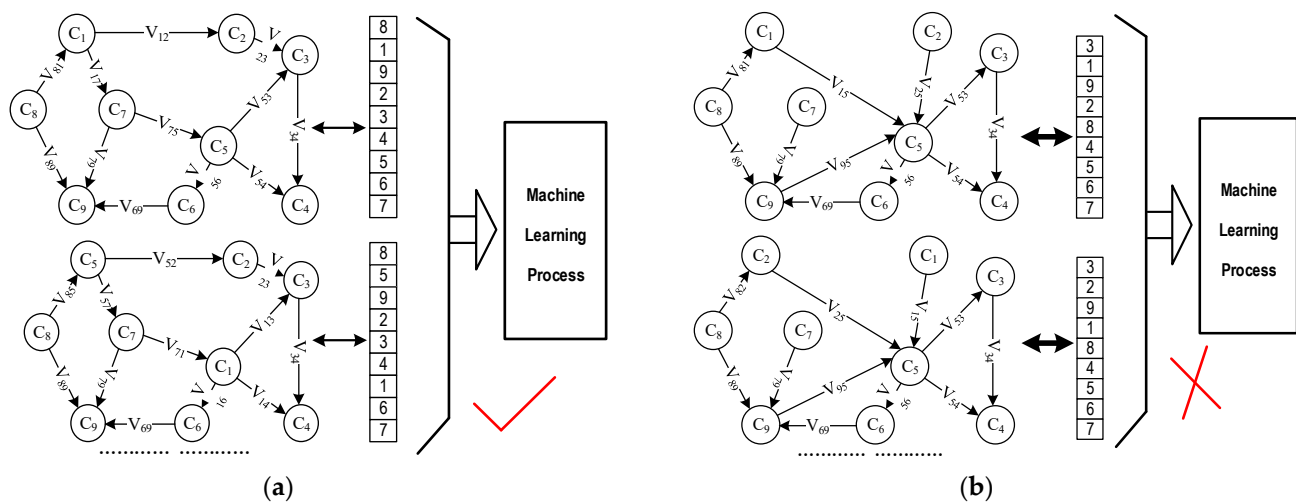


Figure 7. Risk of data augmentation approach: (a) is one application mapping and augmentations; (b) is the other application mapping and augmentations.

Figure 7a is one application mapping and corresponding augmentations with Algorithm 1. Figure 7b is the other application mapping and corresponding augmentations with Algorithm 1. Assuming that the dataset is composed with two applications and corresponding augmentations. Due to the large number of samples generated, the batch size is smaller than the number of samples, which causes that one batch training to happen in one application and its augmentations. If the number of augmentations is several times that of the batch size, network parameters will nearly converge to an unexpected local optimal point and cause the convergence process difficulty in the other application.

Although the above problem can be alleviated through training parameter adjustment, the training problems cannot be completely solved.

In order to avoid this problem, the order of dataset should be changed. Through changing the order of dataset, in a batch, there will be no two augmentations generated by one mapping solution.

A dataset disorder approach is proposed to eliminate the risk mentioned above. The approach is shown below (Algorithm 2):

Algorithm 2 Dataset Disorder

Input: dataset with augmentation, which includes N initial samples and $N \cdot n!$ samples generated. (The initial sample is written as A , and the corresponding sample generated is written as a).

Output: disorder dataset

01: Let the initial sequence $\{A\}$ be $\{1, 2, 3, \dots, N\}$.

02: New dataset's size is K , $k = 0$;

03: Loop begins, $i = 1$,

04: Loop begins, $j = 1$,

05: Randomly sample an element, called P , from $\{A\}$ without replacement.

06: Randomly sample a number, called m , in range $(1, n!)$.

07: Sample the m th sample generating from the P th initial sample into a new dataset.

08: $k = k + 1$

09: If $k = K$, end all Loops

10: If $j = N$, end Loop

11: If $i = n!$, end Loop

2.2.4. Multi-Label Machine Learning Model

Finally, a multi-label machine learning model has been established to fit the dataset proposed, as Figure 8 shows.

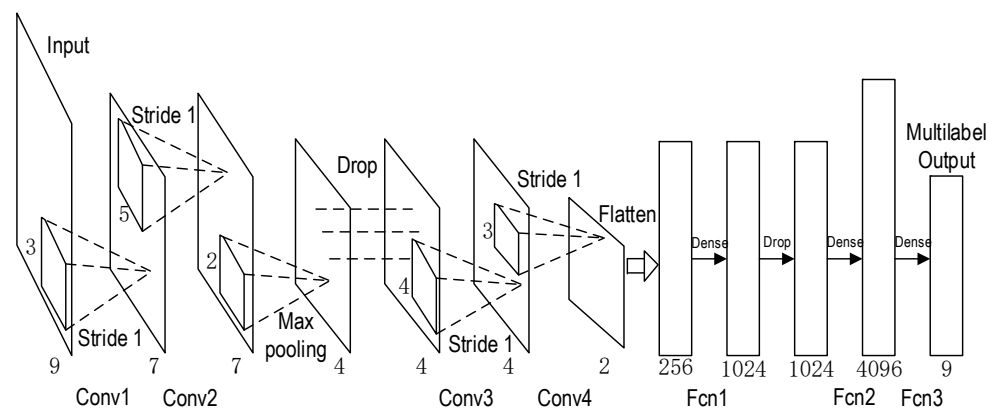


Figure 8. Multi-label machine learning model.

The input of the multi-label machine learning model is the point-to-point task matrix. The output is the core column vector of the nearest neighbor matrix. After combining the core column vector of each core, the mapping sequence could be obtained.

The parameters and outputs for the multi-label machine learning model are as follows.

1. The input layer, the task graph matrix is 9×9 .
 2. The Conv1 layer, performs a convolution operation on the input matrix with a 3×3 size and a convolution kernel with a channel number of 32, with a step size of 1 and no boundary padding.
 3. The Conv2 layer uses $32 \times 5 \times 5$ size convolution kernels, with a step size of 1 and the same boundary padding.
 4. The Max Pooling layer sizes 2×2 with the same boundary padding.
 5. The Dropout layer's rate is set as 0.25.
 6. The Conv3 layer, uses $64 \times 4 \times 4$ size convolution kernels, with a step size of 1 and the same boundary padding.
 7. The Conv4 layer, uses $64 \times 3 \times 3$ size convolution kernels, with a step size of 1 and no boundary padding.
 8. The Flatten layer.
- (The Full Connect layer is written as Fcn layer).
9. The Fcn1 layer output dimension is 1024 with 0.5 dropout rate.
 10. The Fcn2 layer output dimension is 4096.
 11. The Fcn3 layer output dimension is 9 with sigmoid.

3. Simulation and Results

Simulations are performed on Python, Matlab, and TGFF (Task Generate for Free), where TGFF is used to generate tasks and python is used for searching the global optimal solution and establishing the machine learning model. The topologies of NoC are ring and torus. The size of NoC is set as 9. The routing algorithm is 'XY'. Two optimization targets are set. One is the minimum delay, and the other is the minimum energy.

In order to verify the proposed model, two kinds of experiment are designed. One is worked out with task graphs with fixed count connections, which is designed for verifying the model with a regularity task graph. The other is simulated with task graphs with random count connections, which is designed for verifying the universality of proposed model. Experiments are both developed in ring and torus topology.

In the period of preparing the dataset, 4000 original datasets are globally selected, where 2000 are used to train the proposed model and the others are used to test the training accuracy and loss. By using the data augmentation approach, 20,000 samples are generated with 2000 samples which are used to train the proposed model.

The test loss during the training process of two experiments are shown in Figures 9 and 10.

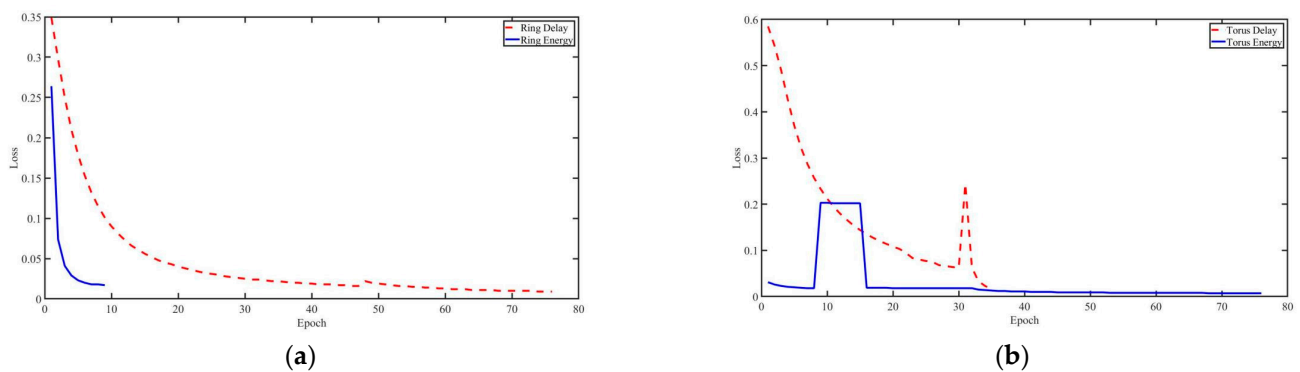


Figure 9. The test loss of NoC with fixed connections number: (a) Ring, (b) Torus.

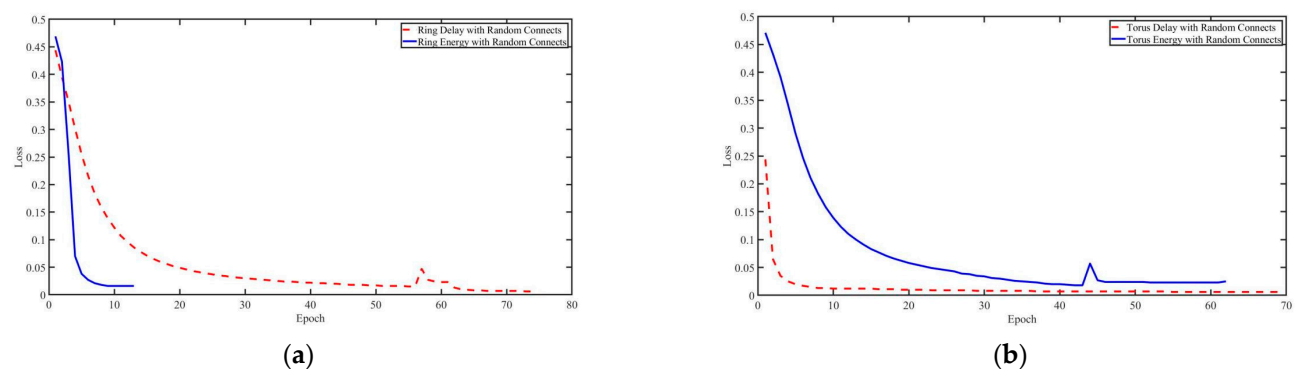


Figure 10. The test loss of NoC with random connections number: (a) Ring, (b) Torus.

Table 1 shows the final test accuracy and test loss of all simulations. The simulation of the Ring with a fixed connection number is written as Ring Fixed. The simulation of the Ring with a random connection number is written as Ring Random. The simulation of the Torus with a fixed connection number is written as Torus Fixed. The simulation of the Torus with a random connection number is written as Torus Random.

Table 1. Test accuracy and test loss.

Performance	Ring Fixed	Ring Random	Torus Fixed	Torus Random
Delay accuracy	99.7%	99.6%	99.8%	99.8%
Delay loss	0.009	0.012	0.006	0.006
Energy accuracy	99.4%	99.8%	99.6%	99.4%
Energy loss	0.017	0.007	0.014	0.018

On the basis of model validation, mapping sequence accuracy and performance distance between predicted samples and correct samples are simulated and calculated.

If the proposed machine learning model predicts correctly, the performance of the global search and the model proposed are the same. If not, the sequence predicted cannot achieve any performance advantage. In order to compare the performance difference between the global search and the model proposed, 200 samples of a test samples dataset are selected for performance calculation.

Tables 2–9 show the performance of wrong task graph mapping samples; the rest are correct and have the same performance with global searching.

Table 2. Fixed connects NoC Application mapping on ring topology (Delay).

Performance	Mapping 1	Mapping 2	Mapping 3	Mapping 4
Best Delay	77,994	76,215	67,481	76,471
False Delay	81,797	97,055	100,036	113,582

Table 3. Fixed connects NoC Application mapping on ring topology (Energy).

Performance	Mapping 1	Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping 6
Best Energy(pj)	16.6191	18.7731	18.9083	19.0964	16.4425	18.8791
False Energy(pj)	18.9333	27.7608	20.3984	28.2356	18.1124	19.3098

Table 4. Random connects NoC Application mapping on ring topology (Delay).

Performance	Mapping 1	Mapping 2	Mapping 3	Mapping 4	Mapping 5
Best Delay	46,474	40,856	76,162	69,169	33,749
False Delay	57,752	43,754	111,034	102,352	34,351

Table 5. Random connects NoC Application mapping on ring topology (Energy).

Performance	Mapping 1	Mapping 2	Mapping 3	Mapping 4	Mapping 5
Best Energy (pj)	15.4384	13.9859	16.1008	20.9854	105,578
False Energy(pj)	21.6159	16.9352	22.4784	27.8658	150,402

Table 6. Fixed connects NoC Application mapping on torus topology (Delay).

Performance	Mapping 1	Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping 6
Best Delay	47,608	26,760	23,987	38,022	26,947	27,646
False Delay	65,645	32,008	26,040	38,627	27,569	39,028
Performance	Mapping 7	Mapping 8				
Best Delay	30,437	25,305				
False Delay	35,262	25,740				

Table 7. Fixed connects NoC Application mapping on torus topology (Energy).

Performance	Mapping 1	Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping 6
Best Energy(pj)	12.7440	6.9261	8.5507	11.1743	7.8398	7.3871
False Energy(pj)	17.4792	9.1960	11.5693	12.7215	8.2205	9.9535
Performance	Mapping 7	Mapping 8	Mapping 9	Mapping 10		
Best Energy(pj)	11.4610	8.2627	12.4991	10.7195		
False Energy(pj)	16.9062	10.0753	17.2832	11.7211		

Table 8. Random connects NoC Application mapping on torus topology.

Performance	Mapping 1	Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping 6
Best Delay	37,962	34,662	58,728	53,211	30,466	48,027
False Delay	47,258	45,863	80,888	71,294	32,943	59,992
Performance	Mapping 7	Mapping 8	Mapping 9			
Best Delay	20,525	63,852	26,871			
False Delay	24,018	70,997	30,298			

Table 9. Random connects NoC Application mapping on torus topology.

Performance	Mapping 1	Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping 6
Best Energy(pj)	29.6195	32.7679	34.8004	35.9488	27.5414	32.5064
False Energy(pj)	36.2185	44.3900	39.6032	47.7238	29.1801	48.1053
Performance	Mapping 7	Mapping 8	Mapping 9	Mapping 10	Mapping 11	Mapping 12
Best Energy(pj)	27.9463	32.8899	28.3429	27.5414	29.6371	33.3850
False Energy(pj)	36.1243	45.2444	35.51304	39.8097	37.7460	35.8770

Table 10 shows the test mapping sequence accuracy and the relative average performance distance (RAPD) of wrong test samples and corresponding correct samples.

Table 10. Test accuracy and performance distance.

	Table 2	Table 3	Table 4	Table 5	Table 6	Table 7	Table 8	Table 9
Accuracy	98%	97%	97.5%	97.5%	96%	95%	94.5%	94%
RAPD	31.6%	22.1%	31.1%	42.4%	17.5%	28.2%	23.8%	27.5%

From Table 10, the test mapping accuracy is lower than the model accuracy. The wrong sample loses its performance advantage and has a large performance gap with the correct mapping sequence, which causes large system performance loss. The average NoC mapping accuracy is 96.3%.

The machine learning mapping algorithms proposed have a good performance in predicting mapping sequence under different optimization targets with ring or torus NoC.

4. Conclusions

In this paper, an NoC machine learning mapping algorithm is established to solve NoC mapping problem. Firstly, the space complexity and symmetry of NoC mapping are analyzed. A label problem is presented with mapping symmetry. Secondly, in order to solve the symmetry problem, a network on a chip mapping dataset with low complexity and no symmetry is proposed for machine learning. The complexity of the dataset changes with the number of ports in a single core of the network on chip. If it is a fully connected network on chip, the proposed method does not reduce the complexity. Thirdly, a data augmentation approach is proposed to establish a machine learning dataset, which includes an augmentation approach and a disorder approach. Finally, a multi-label machine learning model is established and verified with a defined dataset. The simulation results have confirmed that the machine learning mapping algorithm proposed has at least 99.6% model accuracy and an average of 96.3% mapping accuracy to predict optimal solutions in different optimization targets and different topologies. Although the proposed model has high prediction accuracy, it will cause large mapping performance loss in the case of wrong mapping sequence prediction. It is different from the heuristic mapping algorithm, as the heuristic mapping algorithm falls into the local optimal solution and causes performance

loss. The performance of heuristic mapping is better than randomly mapping, however, a wrong mapping sequence obtained by the proposed model can be approximated as a random mapping sequence, which results in greater performance loss. In future work, the method of identifying and dealing with wrongly predicted mapping solutions will be discussed.

Author Contributions: For research articles with several authors, Conceptualization, X.W.; methodology, X.W. and C.X.; validation, X.W.; formal analysis, X.W.; investigation, X.W. and Y.L.; resources, X.L. and Y.L.; data curation, X.W. and C.X.; writing—original draft preparation, X.W., L.Z., S.W. and D.C.; writing—review and editing, X.W. and C.X.; visualization, X.W.; supervision, Y.L. and Y.Y.; project administration, X.L. and Y.L.; funding acquisition Y.L. and C.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Natural Science Foundation of Guangdong, China (no.2021A1515012293), the Opening Project (no. ZHD202006) of Science and Technology on Reliability Physics and Application Technology of Electronic Component Laboratory, National Natural Science Foundation of China Youth Fund under Grant 62004146 and the China Postdoctoral Science Foundation funded project under Grant 2021M692498, Science and Technology Projects in Guangzhou (SL2022A04J00095), the Fundamental Research Funds for the Central Universities (XJSJ23106).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available from the authors upon reasonable request.

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

References

1. Soumya, J.; Tiwary, S.; Chattopadhyay, S. Area-performance trade-off in floorplan generation of Application-Specific Network-on-Chip with soft cores. *J. Syst. Archit.* **2014**, *61*, 1–11. [\[CrossRef\]](#)
2. Dally, W.J.; Towles, B. Route packets, not wires: On-chip interconnection networks. In Proceedings of the 38th Design Automation Conference (IEEE Cat. No.01CH37232), Las Vegas, NV, USA, 22 June 2001; pp. 684–689.
3. Ogras, U.Y.; Bogdan, P.; Marculescu, R. An Analytical Approach for Network-on-Chip Performance Analysis. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2010**, *29*, 2001–2013. [\[CrossRef\]](#)
4. Nedjah, N.; Silva Junior, L.; de Macedo Mourelle, L. Congestion-aware ant colony based routing algorithms for efficient application execution on Network-on-Chip platform. *Expert Syst. Appl.* **2013**, *40*, 6661–6673. [\[CrossRef\]](#)
5. Sahu, P.K.; Shah, T.; Manna, K.; Chattopadhyay, S. Application Mapping Onto Mesh-Based Network-on-Chip Using Discrete Particle Swarm Optimization. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2013**, *22*, 300–312. [\[CrossRef\]](#)
6. Reddy, B.N.K.; Kar, S. Energy Efficient and High Performance Modified Mesh based 2-D NoC Architecture. In Proceedings of the 2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR), Paris, France, 7–10 June 2021; pp. 1–5.
7. Wang, X.; Choi, T.M.; Yue, X.; Zhang, M.; Du, W. An Effective Optimization Algorithm for Application Mapping in Network-on-Chip Designs. *IEEE Trans. Ind. Electron.* **2019**, *67*, 5798–5809. [\[CrossRef\]](#)
8. Mohiz, M.J.; Baloch, N.K.; Hussain, F.; Saleem, S.; Zikria, Y.B.; Yu, H. Application Mapping Using Cuckoo Search Optimization with Lévy Flight for NoC-Based System. *IEEE Access* **2021**, *9*, 141778–141789. [\[CrossRef\]](#)
9. Reddy, B.N.K.; Kar, S. Machine Learning Techniques for the Prediction of NoC Core Mapping Performance. In Proceedings of the 2021 IEEE 26th Pacific Rim International Symposium on Dependable Computing (PRDC), Perth, Australia, 1–4 December 2021; pp. 1–4.
10. Miki, S.; Yamamoto, D.; Ebara, H. Applying Deep Learning and Reinforcement Learning to Traveling Salesman Problem. In Proceedings of the 2018 International Conference on Computing, Electronics & Communications Engineering (iCCECE), Southend, UK, 16–17 August 2018; pp. 65–70.
11. Zhang, R.; Prokhorchuk, A.; Dauwels, J. Deep Reinforcement Learning for Traveling Salesman Problem with Time Windows and Rejections. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–8.
12. Wu, Y.; Song, W.; Cao, Z.; Zhang, J.; Lim, A. Learning Improvement Heuristics for Solving Routing Problems. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 5057–5069. [\[CrossRef\]](#) [\[PubMed\]](#)
13. Zhong, L.; Sheng, J.; Jing, M.; Yu, Z.; Zeng, X.; Zhou, D. An optimized mapping algorithm based on Simulated Annealing for regular NoC architecture. In Proceedings of the 2011 9th IEEE International Conference on ASIC, Xiamen, China, 25–28 October 2011; pp. 389–392.

14. Shen, W.T.; Chao, C.H.; Lien, Y.K.; Wu, A.Y. A New Binomial Mapping and Optimization Algorithm for Reduced-Complexity Mesh-Based On-Chip Network. In Proceedings of the First International Symposium on Networks-on-Chip (NOCS'07), Princeton, NJ, USA, 7–9 May 2007; pp. 317–322.
15. Majd, A.; Sahebi, G.; Daneshtalab, M.; Plosila, J.; Tenhunen, H. Hierarchal Placement of Smart Mobile Access Points in Wireless Sensor Networks Using Fog Computing. In Proceedings of the 2017 25th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), St. Petersburg, Russia, 6–8 March 2017; pp. 176–180.
16. Murali, S.; Micheli, G.D. Bandwidth-constrained mapping of cores onto NoC architectures. In Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, Paris, France, 16–20 February 2004; Volume 892, pp. 896–901.
17. Tang, L.; Kumar, S. A two-step genetic algorithm for mapping task graphs to a network on chip architecture. In Proceedings of the Euromicro Symposium on Digital System Design, Belek-Antalya, Turkey, 1–6 September 2003; pp. 180–187.
18. Upadhyay, M.; Shah, M.; Bhanu, P.V.; Soumya, J.; Cenkeramaddi, L.R. Multi-application Based Network-on-Chip Design for Mesh-of-Tree Topology Using Global Mapping and Reconfigurable Architecture. In Proceedings of the 2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID), Delhi, India, 5–9 January 2019; pp. 527–528.
19. Hsin, H.K.; Chang, E.J.; Su, K.Y.; Wu, A.Y. Ant Colony Optimization-Based Adaptive Network-on-Chip Routing Framework Using Network Information Region. *IEEE Trans. Comput.* **2014**, *64*, 2119–2131. [[CrossRef](#)]
20. Xu, C.; Liu, Y.; Li, P.; Yang, Y. Unified Multi-objective Mapping for Network-on-chip Using Genetic based Hyper-heuristic Algorithms. *IET Comput. Digit. Tech.* **2017**, *12*, 158–166. [[CrossRef](#)]
21. Bhardwaj, K.; Mane, P.S. C3Map and ARPSO based mapping algorithms for energy-efficient regular 3-D NoC architectures. In Proceedings of the Technical Papers of 2014 International Symposium on VLSI Design, Automation and Test, Hsinchu, Taiwan, 28–30 April 2014; pp. 1–4.
22. Singh, R.; Armour, S.; Khan, A.; Sooriyabandara, M.; Oikonomou, G. Heuristic Approaches for Computational Offloading in Multi-Access Edge Computing Networks. In Proceedings of the 2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications, London, UK, 31 August–3 September 2020; pp. 1–7.
23. Zhang, J.; Li, Z.; Pu, Z.; Xu, C. Comparing Prediction Performance for Crash Injury Severity Among Various Machine Learning and Statistical Methods. *IEEE Access* **2018**, *6*, 60079–60087. [[CrossRef](#)]
24. Reddy, M.P.; Aneesh, A.; Praneetha, K.; Vijay, S. Global Warming Analysis and Prediction Using Data Science. In Proceedings of the 2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 11–13 November 2021; pp. 1055–1059.
25. Ying, H.; Heid, K.; Hollstein, T.; Hofmann, K. A genetic algorithm based optimization method for low vertical link density 3-dimensional Networks-on-Chip many core systems. In Proceedings of the NORCHIP 2012, Copenhagen, Denmark, 12–13 November 2012; pp. 1–4.
26. Wu, C.; Deng, C.; Liu, L.; Han, J.; Chen, J.; Yin, S.; Wei, S. An Efficient Application Mapping Approach for the Co-Optimization of Reliability, Energy, and Performance in Reconfigurable NoC Architectures. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2015**, *34*, 1264–1277. [[CrossRef](#)]
27. Hou, J.; Han, Q.; Radetzki, M. A Machine Learning Enabled Long-Term Performance Evaluation Framework for NoCs. In Proceedings of the 2019 IEEE 13th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoc), Singapore, 1–4 October 2019; pp. 164–171.
28. Rao, N.; Ramachandran, A.; Shah, A. MLNoC: A Machine Learning Based Approach to NoC Design. In Proceedings of the 2018 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), Lyon, France, 24–27 September 2018; pp. 1–8.
29. Lin, T.R.; Penney, D.; Pedram, M.; Chen, L. A Deep Reinforcement Learning Framework for Architectural Exploration: A Routerless NoC Case Study. In Proceedings of the 2020 IEEE International Symposium on High Performance Computer Architecture (HPCA), San Diego, CA, USA, 22–26 February 2020; pp. 99–110.
30. Das, S.; Doppa, J.R.; Kim, D.H.; Pande, P.P.; Chakrabarty, K. Optimizing 3D NoC design for energy efficiency: A machine learning approach. In Proceedings of the 2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Austin, TX, USA, 2–6 November 2015; pp. 705–712.
31. Forrest, S. Genetic algorithms: Principles of natural selection applied to computation. *Science* **1993**, *261*, 872–878. [[CrossRef](#)]
32. Weng, X.D.; Liu, Y.; Yang, Y.T. Network-on-chip heuristic mapping algorithm based on isomorphism elimination for NoC optimisation. *IET Comput. Digit. Tech.* **2020**, *14*, 272–280. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.