Crosstalk-Aware Automatic Topology Customization and Optimization for Wavelength-Routed Optical NoCs

Moyuan Xiao, Tsun-Ming Tseng, *Member, IEEE* and Ulf Schlichtmann, *Senior Member, IEEE*

***Abstract*—Optical network-on-chip (ONoC) is an emerging upgrade for electronic network-on-chip (ENoC). As a kind of ONoC, wavelength-routed optical network-on-chip (WRONoC) shows ultra-high bandwidth and ultra-low latency in data communication. Manually designed WRONoC topologies typically reserve all to all links. This causes the waste of resources. Topology customization for each individual communication network can save resources, but requires automation for efficient design. The state-of-the-art design automation method is not efficient and does not support crosstalk analysis and signal-to-noise ratio (SNR) optimization. Moreover, the state of the art does not consider the physical locations of the data sending/receiving ports, causing unavoidable detours and crossings in the physical layout. In this work, we present FAST+: an automatic topology customization and optimization method. Compared with the state of the art, FAST+ operates much more efficiently and proposes a concrete router-level crosstalk-analysis method and a novel SNR optimization algorithm. This work also provides solutions to avoid detours and crossings in the physical layout. When SNR optimization is not enabled, experimental results show that FAST+ runs thousands times faster than the state of the art on average while providing multiple better or equally good topologies regarding resource usage and the worst-case insertion loss. When SNR optimization is enabled, FAST+ provides 1.75× better worst-case SNR on average after the optimization while not sacrificing resource usage and the worst-case insertion loss.**

***Index Terms*—WRONoC, Crosstalk, SNR, Computer-Aided Design, Integer Linear Programming, Optimization.**

I. Introduction

Multiprocessor system-on-chip (MPSoC) is the mainstream solution for applications demanding dense computations. Due to the demands of high-quality communication in MPSoCs, a novel data transmission approach with high bandwidth and low latency is urgently required. In recent years, optical network-on-chip (ONoC) has become a promising next-generation data transmission platform. Instead of using electronic signals, ONoC uses optical signals to transmit data and thus acquires ultra-high bandwidth and ultra-low latency. ONoC can be classified into two kinds: 1) active ONoC in which a control system is applied to control the routing behavior in real time during data transmission. 2) passive ONoC in which the paths of all signals are predefined, also named as wavelength-routed optical network-on-chip (WRONoC) [1] [2]. WRONoCs do not require control systems. Thus, the latency in WRONoCs is even lower than the latency in active ONoCs. Yet, WRONoCs consume more resources because the path of each optical signal has to be reserved. Thus, a design automation tool which can optimize the resource usage and reduce the redundancy is rather important for WRONoCs. In this work, we propose a fully automatic tool which targets the efficient optimization of resource usage, the worst-case insertion loss, and the worst-case SNR for WRONoCs.

WRONoC is enabled by the rapid development of silicon photonics and CMOS fabrication technology. There are two core components in WRONoC: 1) Optical waveguide. It is the medium where light passes through, like conductor for electrons in ENoC. Multiple optical signals with different wavelengths can travel along the same waveguide simultaneously. This is known as wavelength-division multiplexing (WDM) [7]. 2) Silicon Microring Resonator (MRR). An MRR is a ring-formed waveguide. When the length of an MRR's optical path is an integer multiple of an optical signal's wavelength, we say this signal is resonant with the MRR, otherwise nonresonant [8]. As shown in Fig. 1, if an optical signal is resonant with an MRR, it changes its direction when passing by the MRR (also called "drop" at the MRR). If nonresonant, it ignores the MRR and goes straight.

Optical signals suffer power losses when they travel in the network. An optical signal suffers propagation loss, when it travels in the waveguide. When a signal travels through a crossing, it suffers crossing loss. When a signal passes by a nonresonant MRR, it suffers passing loss. But when it drops at a resonant MRR, the signal suffers drop loss. Besides, a signal suffers bending loss when it travels in a bending waveguide. These power losses are generally called insertion loss [9]. The

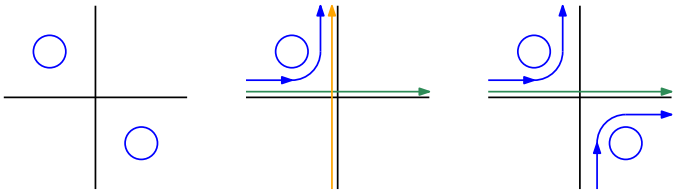


Fig. 1: Basic components and routing behavior of WRONoC. (a) A waveguide crossing with 2 MRRs. (b)(c) Blue signals are resonant with the MRRs. Orange and green signals are nonresonant with the MRRs.

The preliminary version of this paper was published in the Proceedings of the Design, Automation and Test in Europe Conference (DATE), 2021. This work was supported by German Research Foundation (DFG) under Grant 439798838. This article was recommended by Associate Editor L. Carloni. (*Corresponding author: Tsun-Ming Tseng.*)

Moyuan Xiao, Tsun-Ming Tseng and Ulf Schlichtmann are with the Chair of Electronic Design Automation, Technical University of Munich (TUM), Arcisstr. 21, Munich 80333, Germany (e-mail: moyuan.xiao@tum.de; tsun-ming.tseng@tum.de; ulf.schlichtmann@tum.de).

TCAD, DOI: 10.1109/TCAD.2022.3151247
https://ieeexplore.ieee.org/document/9712637

worst-case insertion loss among all signals is an important property because it determines the required laser power for the whole system. A waveguide crossing with one or two MRRs as shown in Fig. 1 is called crossing switching element (CSE). CSE is the basic routing element of most WRONoC topologies.

Crosstalk is another important aspect for WRONoC. It is a small portion of signal power arriving at the wrong destination, namely noises to the desired optical signals [9]. Crosstalk and the insertion loss determine the quality of an optical signal. The quality of an optical signal can be quantified as signal-to-noise ratio (SNR). SNR and SNR in dB are expressed by (1) and (2) respectively, in which $P_S^{\lambda_n}$ is the power of optical signal λ_n, $P_N^{\lambda_n}$ is the unwanted noise power to signal λ_n.

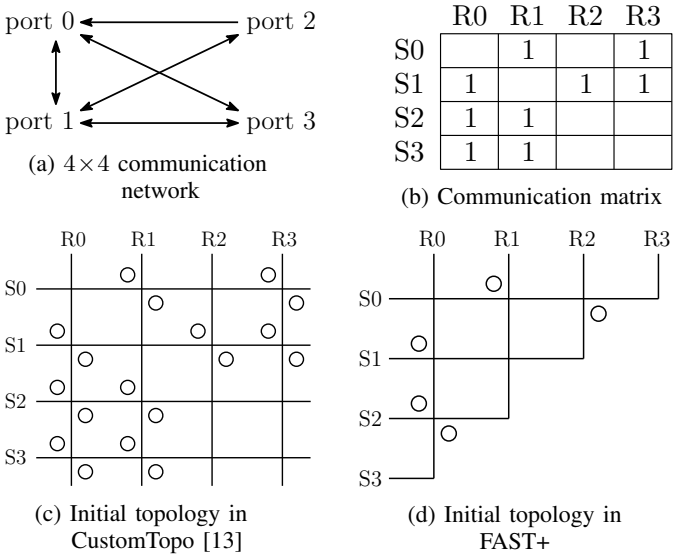
$$SNR^{\lambda_n} = \frac{P_S^{\lambda_n}}{P_N^{\lambda_n}} \tag{1}$$

$$SNR_{dB}^{\lambda_n} = 10\lg\frac{P_S^{\lambda_n}}{P_N^{\lambda_n}} \tag{2}$$

To achieve high SNR, large $P_S^{\lambda_n}$ and small $P_N^{\lambda_n}$ are simultaneously required. In other words, we pursue small insertion loss and small crosstalk in order to achieve high quality optical signals. In this work, we propose a novel crosstalk analysis algorithm which is universal for all WRONoC topologies built based on CSE e.g. folded crossbar [10], lambda router [11], snake [10], and GWOR [12] (Fig. 9). This algorithm can calculate the insertion losses and crosstalk for all the signals in a network accurately. Based on this algorithm, we develop an efficient SNR optimization method.

Fig. 2a shows a 4×4 communication network. In this network, 4 physical ports need to be connected. Each port has a sender (S) and a receiver (R), e.g. port 0 has S0 and R0. If a sender communicates with a receiver, we call it a **communication**, e.g. **communication (S0, R1)**. In the matrix in Fig. 2b, each communication is marked with 1, making it a communication matrix. Typical manually designed WRONoC topologies such as folded crossbar, lambda router or GWOR support full connectivity, i.e. each sender (master) sends messages to all receivers (slaves) and each receiver receives messages from all senders. They should be customized when being applied to application-specific networks in which full connectivity is not required [13]. Otherwise, topologies supporting full connectivity cause the waste of resources and power. The customization on the other hand is tedious, especially for large-scale communication networks. Furthermore, there is no optimization process in manual customization.

So far, there is only one fully automatic topology-synthesis tool realizing topology customization, optimization, and wavelength assignment for WRONoCs, called CustomTopo [13]. CustomTopo includes the WRONoC topology structure and its communications into an integer-linear-programming (ILP) model. The optimization targets are MRR usage, wavelength usage, and the worst-case insertion loss. The aspects that can be improved in CustomTopo are: (1) The initial topology of CustomTopo is a complete matrix. An example is shown in Fig. 2c. This leads to numbers of empty crossings (crossings with no MRRs), especially for sparse communication networks. (2) In CustomTopo, the basic routing element is add-drop filter (ADF). It is a CSE with two MRRs as shown in Fig. 1a. One of the two MRRs is usually redundant and cannot be removed. (3) The computational complexity of the ILP model increases exponentially with the growth of communication density and network size. (4) CustomTopo does not consider the physical port locations of the network. This leads to extra waveguide detours and crossings in the physical layout. (5) CustomTopo does not support crosstalk analysis and SNR optimization.



(a) 4×4 communication network

(b) Communication matrix

(c) Initial topology in CustomTopo [13]

(d) Initial topology in FAST+

Fig. 2: A 4×4 communication network and its initial topologies in CustomTopo and FAST+.

In this work, we present FAST+, a fast automatic sweeping topology customization and optimization method for application-specific WRONoCs. FAST+ solves the five problems of CustomTopo by five features respectively: (1) A half-matrix initial topology with fewer empty crossings (shown in Fig. 2d) is proposed. (2) FAST+ addresses each MRR and ensures that no MRR is redundant. (3) FAST+ combines a fast sweeping technique and an ILP model, which makes it thousands times faster than CustomTopo averagely. (4) Multiple topology variations with different port orders are generated. The variation best matching the physical port locations can be selected as the final topology for layout. (5) FAST+ supports exhaustive crosstalk analysis and effective SNR optimization.

The structure of this paper is shown in the following: Section II lays the background and related work in WRONoC topology synthesis. Section III introduces the initial topology used in FAST+ and the general optimization idea. Section IV provides three methods to solve three problems in order to realize the optimization idea. A proof is presented before each method to support the validity of the method. Section V proposes a comprehensive crosstalk and SNR analysis algorithm which can be applied on all WRONoC topologies built based on CSE such as snake, folded crossbar, lambda router, and GWOR. We use this algorithm on FAST+ to calculate and optimize the worst-case SNR. In Section VI, the optimization processes and the experimental results are presented. In Section VII, we introduce three exclusive features in FAST+ which help us save more resources in the

physical layout. In the last section, Section VIII, we draw the conclusion along with a brief outlook on future work.

II. Background and Related Work

WRONoC topology generation has seen much research interest over the past decade. A well-designed topology can lead optical signals accurately to the correct terminal while saving optical components, wavelengths, and power. [10]–[12] present manually designed WRONoC topologies supporting full connectivity. [3] designs a topology customization method only for the topology presented in [11]. This method has to be implemented manually. [6] introduces design automation into WRONoC topology generation for the first time. [13] is so far the only fully automatic topology customization and optimization tool. It addresses the optimization of MRR usage, wavelength usage, and the worst-case insertion loss.

Crosstalk is an intrinsic issue in WRONoC. It accumulates in the network and degrades the SNR. [4], [5], and [9] find the worst-case SNR through theoretical analysis and report that the noise power exceeds the signal power as the size of the network increases. This indicates that crosstalk has a strong impact on the quality of the signal transmission and is necessary to be optimized. So far, there is no such a work that includes crosstalk analysis and SNR optimization into an automatic topology customization and optimization tool.

FAST [14] proposes a fully automatic topology customization and optimization tool targeting the same optimization targets as [13]. FAST outperforms the previous state of the art [13] and is thousands times faster. In this work, we further integrate a comprehensive crosstalk analysis algorithm into FAST and present FAST+. For the first time in automatic topology synthesis, FAST+ realizes SNR optimization.

III. Initial Topology and General Optimization Idea

A. Logic Scheme of the Initial Topology

The logic scheme of the initial topology of FAST+ is illustrated with a 4×4 communication network in Fig. 3a. Rings with different colors represent MRRs resonant with different wavelengths. The numbers inside each MRR represent a communication. For example, (3, 1) in the ring means sender S3 sends a message to receiver R1 and the optical signal changes direction by MRR (3, 1). Not all crossings are associated with two MRRs because the topology does not support full connectivity. Some communications like (S2, R1) don't rely on MRRs. These communications are called **default communications** in this work. The path of a default communication is called **default path**.

The logic structure used in FAST+ is similar to snake, a WRONoC topology proposed in [10]. The difference is that in snake, senders are placed on the top and receivers on the left side. (In FAST+, senders are placed on the left side and receivers on the top, shown in Fig. 3a.) A comprehensive comparison of the layout efficiency of different ONoC logic topologies under practical physical constraints has demonstrated the superiority of snake. Based on the experimental results in [10], snake outperforms other logic topologies including folded crossbar, lambda router, GWOR, and ORNoC [15] in the physical layout regarding the worst-case insertion loss and power consumption. This supports the logic scheme used in FAST+, as the logic scheme used in FAST+ and snake are equivalent as a topology.

B. General Optimization Idea

We optimize the topology by changing the sequence of senders and receivers. For example: In application-specific WRONoCs, not every port simultaneously sends and receives messages. Some of the senders and receivers are redundant. In Fig. 3a, S1 doesn't send any signal and R3 doesn't receive any signal. We set S1 and R3 as the terminals of a default path (Fig. 3b) and remove that entire default path (Fig. 3c). The general optimization idea is: For a communication network, different topology variations can be generated based on different sender/receiver orders. We sweep through these variations to find the best ones. To realize this idea, we need to describe the logic topologies shown in Fig. 3 mathematically, assign a wavelength to each MRR and design an optimization algorithm. In Section IV, we propose three methods to solve these problems and provide three proofs to support the validity of the methods.

IV. Three Proofs and Three Methods

The following three proofs support FAST+. The first proof theoretically supports the validity of a fast initial topology generation method. This method will be introduced following the first proof. The second proof carries out the wavelength assignment rule. After this proof, an integer-linear-programming (ILP) model is presented to minimize the wavelength usage and assign a wavelength to each MRR. The third proof enables a quick method to select those topologies requiring the minimal number of wavelengths.

A. Proof 1 and Method 1: Generate initial topology

Fig. 4a is a half-matrix initial topology of FAST+ supporting full connectivity. Fig. 4b is a complete-matrix topology. We notice MRRs with the same colors in Fig. 4a are symmetric with respect to the antidiagonal in Fig. 4b. If this is always true, we can easily generate a half matrix through folding the complete matrix along the antidiagonal. This method swiftly generates an initial topology simply through matrix manipulation. The generated topology has no redundant MRRs.

Now, we prove: If a complete matrix like Fig. 4b is folded along the antidiagonal, the two MRRs which overlap are exactly the two MRRs associated to a crossing in Fig. 4a. To prove this, we have to prove: 1) The two MRRs associated to a crossing in Fig. 4a (like MRR (0, 0) and (3, 3)) must be

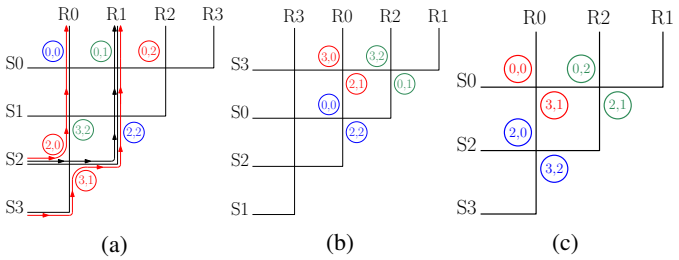


Fig. 3: Logic scheme and an optimization example.

symmetric with respect to the antidiagonal, when they are in a complete matrix. This is proved in Proof 1. 2) Each crossing in a complete-matrix topology is only associated with one MRR. This argument does not need to be proved. It is directly shown in Fig. 4b.

Proof 1: We call the size of the communication matrix "$degree$". For example, in Fig. 4a, the $degree$ of the communication matrix is 4. In general, the default communications of the initial topology can be expressed with $(a, N-a)$ $(N = degree-1, a = 0,1,2,...,N)$. In the 4×4 initial topology, the default communications are $(S0,R3)$, $(S1,R2)$, $(S2,R1)$, $(S3,R0)$, i.e. (0,3), (1,2), (2,1), (3,0). In Fig. 4a, every two default paths have a crossing. If the communications in the upper-left corner and the lower-right corner of a crossing are (p,q) and (m,n), according to the default communication expression, there are always (3):

$$\begin{cases} m = N-q, \\ n = N-p, \quad N,p,q,m,n \in \mathbb{N} \end{cases} \tag{3}$$

Now, we prove: in a complete matrix like Fig. 4b, if two communications (p,q), (m,n) satisfy (3), their MRRs must be symmetric with respect to the antidiagonal. This proof is done in Fig. 4c by proving: 1) The dotted line segment connecting (p,q) and (m,n) is perpendicular to the antidiagonal (the solid line connecting $(0,-N)$ and $(N,0)$). 2) The midpoint of the dotted line segment between (p,q) and (m,n) is on the antidiagonal. In Fig 4c, the gradient of the solid line is 1. (p,q) and (m,n) satisfy (3). As they are now placed in the quadrant IV of a Cartesian coordinate system, we modify (3) into (4):

$$\begin{cases} m = N-|q|, \\ |n| = N-p, \quad N,p,m \in \mathbb{N}, q,n \in \mathbb{Z}, q,n \le 0 \end{cases} \tag{4}$$

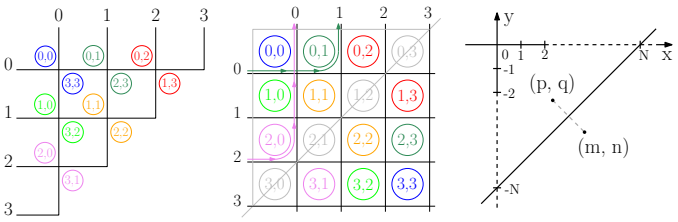
First, using (4), we determine the gradient of the dotted line segment is -1. This proves that the line segment between (p,q) and (m,n) is perpendicular to the antidiagonal. Second, the equation for the antidiagonal in the coordinate system is $y = x-N$. It is easy to prove the midpoint of the line segment between (p,q) and (m,n) is on the antidiagonal. These two arguments prove that any (p,q) and (m,n) satisfying (3) are symmetric with respect to the antidiagonal in a complete matrix.

Method 1: Now, a fast topology customization method based on this proof is proposed using the 4×4 communication network in Fig. 2 as an example:

1) First of all, we generate a communication dictionary shown in (5), to clarify which sender or receiver is represented by which index number in the matrix:

$$\begin{cases} sender\ dict.: & (0{:}S0,\ 1{:}S1,\ 2{:}S2,\ 3{:}S3) \\ receiver\ dict.: & (0{:}R0,\ 1{:}R1,\ 2{:}R2,\ 3{:}R3) \end{cases} \tag{5}$$

2) As shown in Fig. 5a, based on (5), we generate a complete matrix similar to Fig. 2b. Communication nodes on the left side of the antidiagonal and on the antidiagonal are marked with 1, but communication nodes on the right side of the antidiagonal are marked with 2.
3) As shown in Fig. 5b, mirror this complete matrix generated in step 2 with respect to the antidiagonal.
4) As shown in Fig. 5c, add the two matrices shown in Fig. 5a and Fig. 5b. This step overlaps those MRRs which are symmetric with respect to the antidiagonal in a complete matrix.
5) As shown in Fig. 5d, remove non-zero values under the antidiagonal. In Fig. 5d, each 1 represents an MRR in the upper-left corner of a crossing; 2s on the antidiagonal represent default communications; each 2 off the antidiagonal represents an MRR in the lower-right corner of a crossing; 3 means both MRRs are required. The coordinates of these values are called **non-zero coordinates**. Fig. 5d is called **initial matrix**. It can be directly transferred to the initial topology (Fig. 2d).



(a) (b) (c)

Fig. 4: (a) Half-matrix topology. (b)(c) Complete-matrix topology and its coordinate system expression.

B. Proof 2 and Method 2: Wavelength assignment

The basic rule for conflict-free communication in WRONoCs is: A sender must use different wavelengths to send signals to different receivers. A receiver needs to receive different wavelengths from different senders [13]. In the logic scheme of FAST+, this rule can be formulated as this expression: One signal drops maximal one time, and it always drops at the correct position. We simplify this expression into: Wavelengths assigned to each non-zero coordinate on a default path should be different from each other. This is a significant simplification. If this simplification is valid, we do not have to consider every signal in the network, but only the default paths. For the logic scheme used in FAST+, no matter how many signals travel in the network, there are only number of "$degree$" default paths. **Now, we prove** that the simplified rule satisfies the expression.

Proof 2: Fig. 6a shows a 4×4 topology in matrix form. Fig. 6b shows the same topology in graphic form. In Fig. 6a, the simplified rule is implemented: Colors represent different wavelengths. The gray lines indicate default paths. The colors of each non-zero coordinate on a default path are different from each other. In the graphic form, Fig. 6b, there are three kinds of communications: (1) Communication (1, 1) is the first situation. The signal drops at the MRR in the upper-left corner of a crossing. This signal travels both on default path

	0	1	2	3
0	0	1	0	1
1	1	0	1	2
2	1	1	0	0
3	1	2	0	0

(a)

	0	1	2	3
0	0	0	2	1
1	0	0	1	0
2	2	1	0	1
3	1	1	1	0

(b)

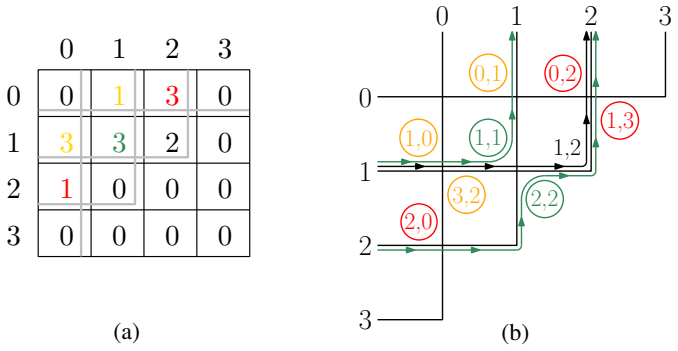
	0	1	2	3
0	0	1	2	2
1	1	0	2	2
2	3	2	0	1
3	2	3	1	0

(c)

	0	1	2	3
0	0	1	2	2
1	1	0	2	0
2	3	2	0	0
3	2	0	0	0

(d)

Fig. 5: Initial matrix generation process.



(a) (b)

Fig. 6: (a) Topology in matrix. (b) Topology in graphic.

(1, 2) and default path (2, 1). On these two default paths, there are no green MRRs except MRR (1, 1). This guarantees that the optical signal drops only one time and drops at MRR (1, 1). (2) Communication (2, 2) represents a signal drops in the lower-right corner of a crossing. This is similar to the first kind. (3) For default communications like communication (1, 2), there are no black MRRs on its optical path. Thus, this signal travels along the waveguide to receiver 2. Based on the analysis of the three kinds of communications, we have proven that the simplified rule satisfies the expression.

Method 2: Based on the communication rule, we present an ILP model to determine the minimal wavelength usage and assign a wavelength to each MRR. The **input** of this ILP model is an initial matrix shown in Fig. 5d. The **outputs** are the minimal wavelength usage and the assignment of a wavelength to each MRR. The **optimization objective** is to minimize the wavelength usage.

The ILP model includes three groups of variables and four constraints. **The first variable group** is $indicator_{(m,n),w}$. They are binary variables. They indicate whether non-zero coordinate (m,n) in the initial matrix takes wavelength w. If (m,n) is assigned with w, $indicator_{(m,n),w}=1$. If (m,n) is not assigned with w, $indicator_{(m,n),w}=0$. **The second variable group** is $W_{(m,n)}$. They are integer variables with a lower bound 1 and an upper bound $degree$. They indicate the wavelength type of (m,n). **The third variable** is W_{max}. It's also an integer variable with a lower bound 1 and an upper bound $degree$. Constraints are listed in the following:

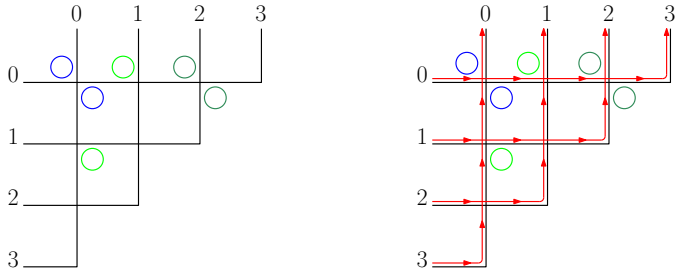
1) Each non-zero coordinate in the initial matrix is assigned with exactly one wavelength:
$$\forall(m,n)\in C:\sum_{w=1}^{degree} indicator_{(m,n),w}=1 \tag{6}$$
C is the set of all non-zero coordinates in an initial matrix.
2) In each default path, a wavelength type must not appear more than once:
$$\forall p_{default}\in P_{default}\,\forall w\in[1,2,...,degree]:\sum_{(m,n)\in nzc} indicator_{(m,n),w}\le 1 \tag{7}$$
$p_{default}$ is one of the default paths in the initial matrix. $P_{default}$ is the set of all default paths in the initial matrix. nzc is the set of all non-zero coordinates on a default path. This constraint describes the communication rule.
3) If $indicator_{(m,n),w}=1$, $W_{(m,n)}$ must be equal to w:
$$\forall(m,n)\in C\,\forall w\in[1,2,...,degree]:\ indicator_{(m,n),w}=1\rightarrow W_{(m,n)}=w \tag{8}$$
This constraint assigns a wavelength to each non-zero coordinate.
4) Finally, we introduce the following constraint:
$$\forall(m,n)\in C: W_{max}\ge W_{(m,n)} \tag{9}$$
To minimize wavelength usage, we just have to minimize the biggest wavelength type number, which is W_{max}.

The runtime of an ILP model increases exponentially with the number of variables and constraints in the model. Thus, in this work, we limit the size of the ILP model and use ILP only to assign wavelengths and to optimize the wavelength usage.

C. Proof 3 and Method 3: The indication of the minimal wavelength usage

We try to directly recognize the topologies requiring the minimal wavelengths without running the ILP model. To do this, we need to find an indication of the wavelength usage. In Proof 2, we have verified that the communication rule for FAST+ is each non-zero coordinate on a default path should be assigned with a different wavelength. If $N_{p_{default}}$ $(p_{default}\in P_{default})$ represents the number of non-zero coordinates on a default path, the minimal wavelength usage is at least $max(N_{p_{default}})$ $(p_{default}\in P_{default})$. We call this number N_{max}. **Now, we prove** that N_{max} is a strong indication of the minimal wavelength usage using Vizing's theorem [16] in graph theory.

Proof 3: As shown in Fig. 7a, in the logic scheme of FAST+, every two default paths have one and only one crossing. Based on graph theory, if default paths in FAST+ are regarded as vertices, and crossings with one or two MRRs are regarded as edges, the topology can be transformed into a simple undirected graph. In this manner, in Fig. 7a, crossing (0, 0) and crossing (1, 0) are edges connecting default path (3, 0) with default path (0, 3) and default path (1, 2), respectively. In this simple undirected graph, if there is no default communication, N_{max} is the maximum degree Δ. To satisfy the communication rule, we only have to edge color the graph. If there are some default communications, N_{max} or $N_{max}-1$ is Δ. To satisfy the communication rule, we not only have to edge color the graph, but also have to reserve wavelengths for the default communications.



(a) (b)

Fig. 7: (a) Topology example. (b) All the default paths can use the same red wavelength without violating the communication rule.

Thus, according to Vizing's theorem, if there is no default communication, N_{max} or $N_{max}+1$ is the minimal wavelength usage. If there are default communications, $N_{max}-1$, N_{max} or $N_{max}+1$ is enough to edge color the graph. But at most one extra wavelength is required for default communications. This is because all the default communications can use this extra wavelength (Fig. 7b). In conclusion, the minimal wavelength usage is at most $N_{max}+2$.

Method 3: In the optimization process, we select the topologies with the smallest N_{max}. Then we use the ILP model to determine the minimal wavelength usage and assign a wavelength to each MRR only for the selected topologies. This method vastly accelerates the algorithm.

V. Systematic Crosstalk Analysis

Crosstalk is a common problem in data transmission. In ONoCs, crosstalk is the noises caused by leakage of non-ideal routing components, or coupling between optical signals and non-ideal routing components. It can be quantified by (10). $P_{in}^{\lambda_n}$ is the power of the input signal λ_n. $P_{crosstalk}^{\lambda_n}$ is the power of the crosstalk noise generated by the signal λ_n. The relation between crosstalk and the wanted optical signal is similar to the relation between the background noises and the radio. When noises are too loud, the valid signal is hard to be distinguished. In WRONoCs, crosstalk is a severe problem. If the topology is not optimized, the power of crosstalk could be bigger than the power of the valid signal, especially for communication networks with large sizes [9].

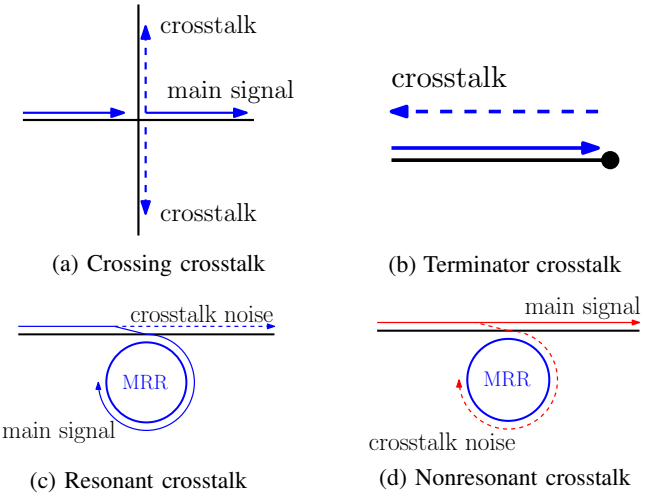
$$Crosstalk_{dB}^{\lambda_n} = 10\lg \frac{P_{in}^{\lambda_n}}{P_{crosstalk}^{\lambda_n}} \tag{10}$$

In WRONoCs, optical signals suffer insertion loss during transmission. This means the power of the wanted signal decreases when passing through waveguides, MRRs or crossings. Insertion loss and crosstalk together cause the low quality of the wanted optical signals. The quality of the signals arriving at the receiver side can be quantified with SNR, which is mathematically expressed in (1) and (2). To achieve high SNR, namely high quality signals, small insertion loss and small crosstalk are simultaneously required. In previous sections, we have designed a redundancy-free initial topology and a special sweeping technique to optimize the worst-case insertion loss. These approaches target low insertion losses. In this section, we analyze crosstalk and optimize SNR. **A universal crosstalk analysis algorithm** is presented to quantify crosstalk and SNR for WRONoC topologies built based on CSE (Fig. 9).

A. Classification for Crosstalk

In this subsection, we introduce different types of crosstalk in silicon photonics and clarify which types of crosstalk are taken into consideration in the algorithm.

Due to the nonideality of the routing components in the WRONoC router, crosstalk is created when an optical signal passes by a crossing or an MRR. Besides, when a signal is terminated by an optical terminator, a small portion of crosstalk will be reflected back [9]. Different types of crosstalk are shown in Fig. 8. (1) Fig. 8a shows crossing crosstalk. When an optical signal travels through a crossing, two portions of crosstalk will be generated to the perpendicular waveguide. (2) Fig. 8b shows terminator crosstalk. It is the reflection of the terminated signal. (3) Fig. 8c shows resonant crosstalk. When an optical signal is resonant with a certain MRR, the main signal drops at the MRR while a small portion of power escapes from the MRR and becomes resonant crosstalk. (4) Fig. 8d shows nonresonant crosstalk. When an optical signal does not resonant with an MRR, it should ignore the MRR and go straight. But due to nonideality, a small portion of power still drops by the MRR and becomes crosstalk noise.



(a) Crossing crosstalk (b) Terminator crosstalk

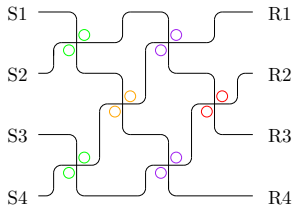
(c) Resonant crosstalk (d) Nonresonant crosstalk

Fig. 8: Types of crosstalk.

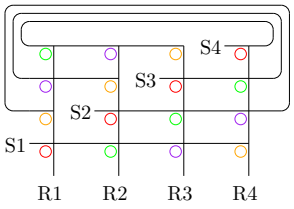
In this work, we only analyze crossing crosstalk, resonant crosstalk, and nonresonant crosstalk. Terminator crosstalk is not considered because the topology proposed in this work does not have optical terminators. Moreover, crosstalk can be classified based on other factors. These classifications are important for the crosstalk analysis. They are listed as follows:

- **Interchannel crosstalk** is the crosstalk whose wavelength is sufficiently different from the desired signal [9].
- **Intrachannel crosstalk** is the crosstalk whose wavelength is the same as or close to the desired signal [9].
- **Incoherent crosstalk**: In physics, two wave sources are coherent if they have identical frequency, waveform, and constant phase difference. If any of these properties is not satisfied, the waves are incoherent. Incoherent crosstalks usually have different wavelengths or come from different sources [19].
- **Coherent crosstalk**: In WRONoCs, if one optical signal generates two crosstalks, these crosstalks are coherent. Thus, they have identical frequency, waveform, and constant phase difference.
- **First-order crosstalk** is the crosstalk which is directly generated by valid optical signals [19].
- **Second/third...-order crosstalk** is the crosstalk generated by other crosstalk. In this work, we only consider first-order crosstalk because the second/third...-order crosstalk has a negligible amount of power.

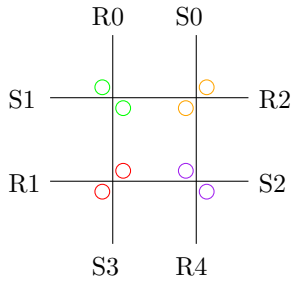
An optical signal usually suffers crosstalks generated by different sources. These crosstalks can be added together to calculate the total noise power and the corresponding SNR for the signal [20]. In general, we analyze first-order incoherent intra/interchannel crosstalk [9].



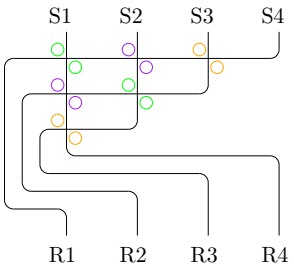
(a) 4×4 Lambda Router



(b) 4×4 Folded Crossbar



(c) 4×4 GWOR



(d) 4×4 Snake

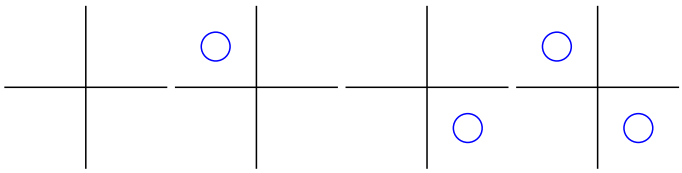
Fig. 9: Representative WRONoC topologies. Circles in different colors represent MRRs resonant with different wavelengths.

B. Formal Crosstalk Analysis

In this subsection, we propose **a general crosstalk analysis algorithm** for all WRONoC topologies which have a basic crossing switching element (CSE) structure. This algorithm is introduced based on the logic scheme of FAST+. As shown in Fig. 9, the representative WRONoC topologies all have the basic CSE structure. It is built by a waveguide crossing and one or two MRRs. Whenever an optical signal passes through a crossing, crosstalk is generated. Crosstalk then travels like a normal signal in waveguides [9]. This makes the analysis of crosstalk very complicated. In this work, we consider crosstalk as a small portion of optical signal and calculate insertion losses, crosstalk, and SNR all in one algorithm.

In FAST+, the CSE structure includes 4 different variations (Fig. 10) i.e. **empty crossings**; **crossings with an MRR in the upper-left corner**; **crossings with an MRR in the lower-right corner**, and **crossings with two MRRs**. Now, we use the logic scheme of FAST+ as an example to propose a special data structure and the details of this crosstalk analysis algorithm.

Fig. 11a shows the logic scheme of FAST+. Senders are placed on the left side. Receivers are placed on top. We separate each waveguide crossing in the topology as a **communication block** with two input ports and two output ports (Fig. 11b and Fig. 11c). As shown in Fig. 11b, for a 4 × 4 communication network, there are 6 communication blocks. To calculate insertion loss and crosstalk all in one step, we need to connect each communication block and let data flow in the topology. Next, we design a special data structure to store the information of signals and crosstalks.



(a) (b) (c) (d)

Fig. 10: CSE variations.

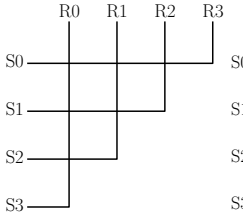
Each communication block has 2 input ports (down and left) and 2 output ports (up and right). The desired signals and the unwanted crosstalk noises are treated equally. They go into a communication block from the left and down sides as inputs, and then go through the internal structure of different communication blocks. Finally, the signals, crosstalks, and newly generated crosstalks are stored in the up and right output ports. The data structure of a communication block is shown in the following:

Inside of a communication block (m, n):

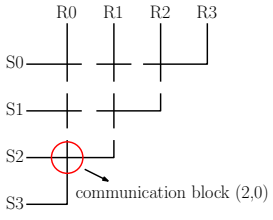
- Inputs:
  - Left:
    - Signals: Signal 1, Signal 2...
    - Crosstalks: Crosstalk 1, Crosstalk 2...
  - Down:
    - Signals: Signal 1, Signal 2...
    - Crosstalks: Crosstalk 1, Crosstalk 2...
- Outputs:
  - Right:
    - Signals: Signal 1, Signal 2...
    - Crosstalks: Crosstalk 1, Crosstalk 2...
  - Up:
    - Signals: Signal 1, Signal 2...
    - Crosstalks: Crosstalk 1, Crosstalk 2...

We identify each communication block with its matrix index (m,n). As shown in Fig. 11b, the communication block with a red circle is identified with matrix index $(2,0)$. Different signals and crosstalks are distinguished by their wavelengths. For example: $(2,0)[down][signals][3]$ is one of the signals on the downside of block $(2,0)$. Its wavelength is marked with 3. $(3,1)[right][crosstalk][4]$ is one of the crosstalks on the right side of block $(3,1)$. Its wavelength is marked with 4. Now, we should connect the communication blocks shown in Fig. 11b. As shown in Fig. 12, we classify all the communication blocks into 4 groups:

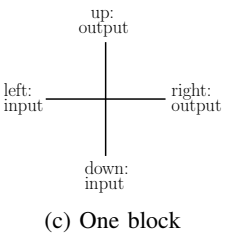
1) The **red** block: Its left side and downside directly connect with external inputs. This block is the starting point of the whole algorithm.
2) The **blue** blocks: Their left inputs are external inputs. Their down inputs connect with the up outputs of the blocks below them. This can be expressed with (11). "=" means the data stored in the "down" of block $(m,0)$ is the same as the data stored in the "up" of block $(m+1,0)$, because they are directly connected.



(a) Logic structure



(b) Comm. blocks



(c) One block

Fig. 11: Communication block separation.

$$(m,0)[down]=(m+1,0)[up],\ m\in[0,1,...,degree-3] \tag{11}$$

3) With the same principle, we write down (12) and (13) for the **orange** blocks and the **green** blocks, respectively:

$$\begin{cases}(m,n)[left]=(m,n-1)[right],\\(m,n)[down]=(m+1,n-1)[right],\\m\in[0,1,...,degree-3],\ n\in[1,2,...,degree-2]\end{cases} \tag{12}$$

$$\begin{cases}(m,n)[left]=(m,n-1)[right],\\(m,n)[down]=(m+1,n)[up],\\m\in[0,1,...,degree-4],\ n\in[1,2,...,degree-3]\end{cases} \tag{13}$$

For topologies in Fig. 9, the principles to classify different communication block groups and connect them are similar. With (11), (12), (13), all communication blocks are connected. The signals and crosstalks can flow inside the topology. In the crosstalk analysis algorithm, the calculation sequence is very important. This is because the outputs of one block are the inputs of other blocks. In a 4×4 network like Fig. 12, the calculation sequence must be (14). In (14), the combinations of numbers represent the matrix index of the communication blocks in Fig. 12.

$$(2,0)\rightarrow(1,0)\rightarrow(0,0)\rightarrow(1,1)\rightarrow(0,1)\rightarrow(0,2) \tag{14}$$

Fig. 3a shows a communication network which has redundant senders/receivers, i.e. empty default paths. In FAST+, empty default paths are systematically cleared out and placed on the left side of the topology (Fig. 3b). In the physical layout, these empty default paths can be directly ignored (Fig. 3c). If a communication network has empty default paths, (11), (12), (13) should be changed according to the number of empty default paths.

Next, we analyze the computation process inside each communication block. Table I summarizes the insertion loss and crosstalk values that are required in this algorithm [9] [21]. In this topology synthesis tool, propagation loss and bending loss are not counted as they cannot be estimated in topology design [13]. There are 4 kinds of communication block (Fig. 12). For each kind, there are 0, 1 or 2 MRRs associated (Fig. 10). In total, 16 different cases need to be analyzed.

To start the algorithm, we clarify the initial input values for the red crossing and the blue crossings shown in Fig. 12. The values of the signals in the external inputs are 0 dB. This means the initial signals have 100% power. On the other

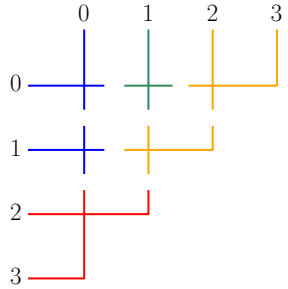


Fig. 12: Different block groups.

TABLE I: Insertion loss and crosstalk values.

Crosstalk/Insertion loss Types	Abbreviation	Value
crossing crosstalk	cc	40 dB
terminator crosstalk	tc	50 dB
resonant crosstalk	rc	25 dB
nonresonant crosstalk	nc	35 dB
drop loss	dl	0.5 dB
crossing loss	cl	0.04 dB
passing loss	pl	0.005 dB

hand, no crosstalk is generated yet. So the space reserved for crosstalk is empty. This is shown in the following:

Inputs of the red crossing:

- Left:
  - Signals: Signal 1 = 0 dB, Signal 2 = 0 dB,...
  - Crosstalks: empty
- Down:
  - Signals: Signal 1 = 0 dB, Signal 2 = 0 dB,...
  - Crosstalks: empty

Inputs of the blue crossings:

- Left:
  - Signals: Signal 1 = 0 dB, Signal 2 = 0 dB,...
  - Crosstalks: empty
- Down:
  - Signals: Signal 1, Signal 2...
  - Crosstalks: Crosstalk 1, Crosstalk 2...

Given a topology as input, this algorithm firstly recognizes which color a crossing (a block) should be assigned with (Fig. 12), then it analyzes insertion loss and crosstalk based on how many MRRs are associated with the crossing (Fig. 10). Now, we propose a general analysis process for all 4 kinds of crossings shown in Fig. 10. As explained, crosstalk is not generated yet in some input ports. So the following process needs to be tailored for different cases. **For crossing crosstalk, we only consider the portions to the right and up, because only crosstalk to the right and up arrives at the receivers.** Crossing crosstalk to the left and down always arrives at the senders, because MRRs are placed only in the upper-left and lower-right corner in FAST+. This is illustrated in Fig. 13.

The index of a crossing is always (m,n). In the following figures, solid lines represent signals, dashed lines represent

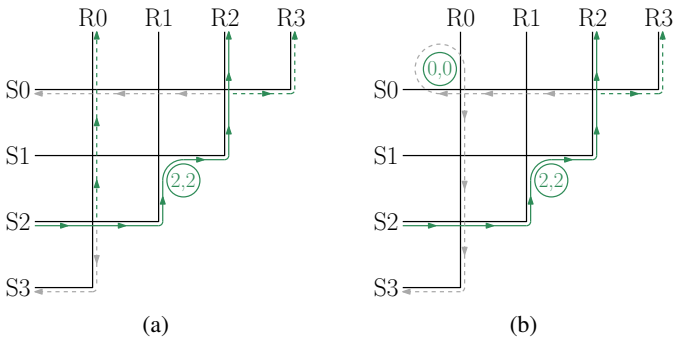


Fig. 13: Crossing crosstalks to the up and right (the green dashed lines) arrives at the receivers. Crossing crosstalks to the left and down (the gray dashed lines) do not reach the receivers. (a) The gray crossing crosstalks directly arrive at the senders. (b) The gray crossing crosstalk arrives at the sender after dropping by MRR $(0,0)$.

crosstalks. If the color of a signal or crosstalk is the same as the MRR, then the signal or the crosstalk is resonant with the MRR, otherwise nonresonant. Nonresonant signals are classified into two kinds: **the nearest nonresonant signals** [21] generate nonresonant crosstalk when passing by a nonresonant MRR. The nearest nonresonant signal means the nonresonant signal whose wavelength is the closest to the wavelength of the resonant signal.; **the other nonresonant signals** do not generate nonresonant crosstalk because their wavelengths are considered too far away from the resonant wavelength. By considering nonresonant crosstalk, we calculate crosstalk more accurately. Now, we introduce the core algorithm about how to analyze the insertion losses and crosstalk for different kinds of crossings as shown in Fig. 10, i.e. **Empty crossing**; **Crossing with an MRR in the upper-left corner**; **Crossing with an MRR in the lower-right corner**; **Crossing with two MRRs**.

(1) Empty crossing:

Fig. 14a shows that an optical signal from the left input port travels through a crossing. The signal itself suffers a crossing loss. A crossing crosstalk is generated to up. These can be expressed with (15). "cl" and "cc" are abbreviations from Table I.

$$\begin{cases} (m,n)[right][signal] = (m,n)[left][signal] - cl, \\ (m,n)[up][crosstalk] = (m,n)[left][signal] - cc \end{cases} \tag{15}$$

Fig. 14b shows an optical signal coming from the down input port. The process is similar with Fig. 14a.

Fig. 14c shows that a crosstalk from the left input port travels through a crossing. The crosstalk suffers a crossing loss. No crossing crosstalk is generated because we only consider first-order crosstalk. This can be expressed with (16):

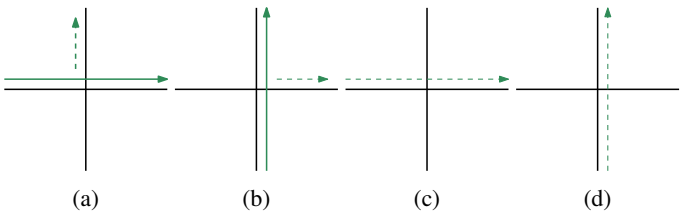
$$(m,n)[right][crosstalk] = (m,n)[left][crosstalk] - cl \tag{16}$$

Fig. 14d is similar with Fig. 14c.

(2) Crossing with an MRR in the upper-left corner:

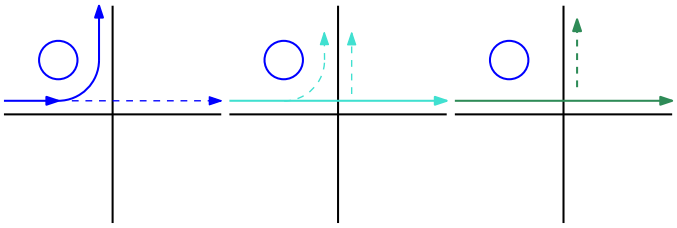
Fig. 15a shows a resonant signal from the left input port drops at the MRR and goes to up. The signal itself suffers a drop loss. A resonant crosstalk is generated to the right of the crossing and suffers a crossing loss. These can be expressed with (17):

$$\begin{cases} (m,n)[up][signal] = (m,n)[left][signal] - dl, \\ (m,n)[right][crosstalk] = (m,n)[left][signal] - rc - cl \end{cases} \tag{17}$$



(a) (b) (c) (d)

Fig. 14: Empty crossing. (a) Signal from left travels to right and generates a crossing crosstalk to up. (b) Signal from down travels to up and generates a crossing crosstalk to right. (c) Crosstalk from left travels to right. (d) Crosstalk from down travels to up.



(a) (b) (c)

Fig. 15: Signals from the left. (a) Resonant signal from left drops to up and generates a resonant crosstalk to right. (b) Nearest nonresonant signals from left travel to right and generate a nonresonant crosstalk and a crossing crosstalk to up. (c) Other nonresonant signals from left travel to right and generate a crossing crosstalk to up.

Fig. 15b shows a nearest nonresonant signal from the left input port travels to the right. The signal itself suffers a passing loss and a crossing loss.

In the meantime, a nonresonant crosstalk and a crossing crosstalk are generated to the up of the crossing. These two crosstalks are from the same source and thus are coherent. We add them and get one crosstalk (18):

$$\begin{cases} (m,n)[up][crosstalk] = 10\lg(10^a + 10^b), \\ a = \frac{(m,n)[left][signal] - nc}{10}, \\ b = \frac{(m,n)[left][signal] - pl - cc}{10} \end{cases} \tag{18}$$

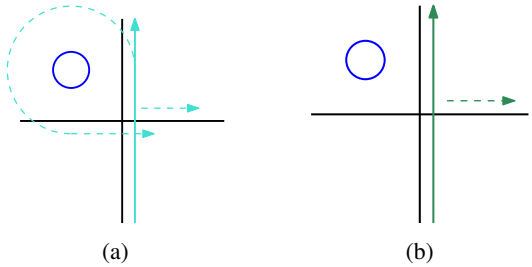
Fig. 15c shows a nonresonant signal from the left input port travels to the right. The signal itself suffers a passing loss and a crossing loss. It also generates a crossing crosstalk to up.

Fig. 16a shows the nearest nonresonant signal from the down input port travels to up. The signal itself suffers a crossing loss and a passing loss. It also generates a crossing crosstalk to the right and a nonresonant crosstalk to the right. The nonresonant crosstalk suffers a crossing loss. These can be expressed with (19) and (20):

$$(m,n)[up][signal] = (m,n)[down][signal] - cl - pl \tag{19}$$

$$\begin{cases} (m,n)[right][crosstalk] = 10\lg(10^a + 10^b), \\ a = \frac{(m,n)[down][signal] - cc}{10}, \\ b = \frac{(m,n)[down][signal] - cl - nc - cl}{10} \end{cases} \tag{20}$$

Fig. 16b shows other nonresonant signals from the down input port travels to up. The signal itself suffers a crossing



(a) (b)

Fig. 16: Signals from down. (a) Nearest nonresonant signals from down travel to up and generate a crossing crosstalk and a nonresonant crosstalk to right. (b) Other nonresonant signals from down travel to up and generate a crossing crosstalk to the right.

loss and a passing loss. It also generates a crossing crosstalk to right. No nonresonant crosstalk is generated because the signal in Fig. 16b is not the nearest nonresonant signal.

For a crossing with an MRR in its upper-left corner, it is impossible that a resonant signal appears in its down input port. This is because in the logic topology of FAST+, one MRR is only responsible for one optical signal. If the lower-right corner of a crossing is empty, it means there is no resonant signal coming from down.

Fig. 17a shows a resonant crosstalk from the left input port drops by an MRR and goes to up. It suffers a drop loss.

Fig. 17b shows a nonresonant crosstalk from the left input port travels through a crossing to the right. The crosstalk suffers a passing loss and a crossing loss.

Fig. 18a shows a resonant crosstalk from the down input port drops by an MRR and goes to the right. It suffers a crossing loss, a drop loss, and then again a crossing loss.

Fig. 18b shows a nonresonant crosstalk from the down input port travels to up. It suffers a crossing loss and a passing loss.

(3) Crossing with an MRR in the lower-right corner:

This case is similar with the last case: Crossing with an MRR in the upper-left corner. Only directions are different. Thus, we skip the figures and equations.

(4) Crossing with two MRRs:

Fig. 19a shows a resonant signal from the left input port drops by the upper-left MRR and goes to up. The signal itself suffers a drop loss. A resonant crosstalk is generated to the right. It is then dropped by the two MRRs and finally dissipated. This can be expressed with (21).

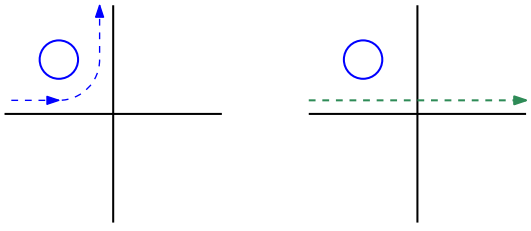
$$(m,n)[up][signal] = (m,n)[left][signal] - dl \tag{21}$$

Fig. 19b shows the nearest nonresonant signal from the left input port travels to the right. The signal itself suffers a passing loss, a crossing loss, and a passing loss. This can be expressed with (22):

$$(m,n)[right][signal] = (m,n)[left][signal] - pl - cl - pl \tag{22}$$

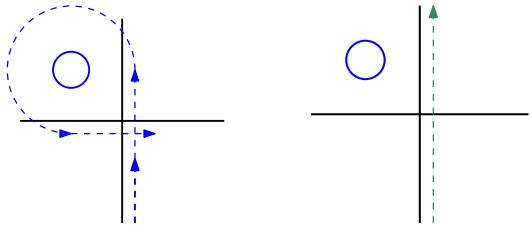
In the meantime, a nonresonant crosstalk, a crossing crosstalk, and a nonresonant crosstalk are generated to up of the crossing. These three crosstalks are from the same source and thus are coherent. We add them and get one crosstalk (23):

$$\begin{cases} (m,n)[up][crosstalk] = 10\lg(10^a + 10^b + 10^c), \\ a = \frac{(m,n)[left][signal] - nc}{10}, \\ b = \frac{(m,n)[left][signal] - pl - cc}{10}, \\ c = \frac{(m,n)[left][signal] - pl - cl - nc - cl - pl}{10} \end{cases} \tag{23}$$



(a) (b)

Fig. 17: Crosstalks from left. (a) A resonant crosstalk from left drops to up, suffering a drop loss. (b) A nonresonant crosstalk from left travels to right, suffering a passing loss and a crossing loss.



(a) (b)

Fig. 18: Crosstalks from down. (a) A resonant crosstalk from down drops to right, suffering a crossing loss, a drop loss, and a crossing loss. (b) A nonresonant crosstalk from down travels to up, suffering a crossing loss and a passing loss.

Fig. 19c shows other nonresonant signals from the left input port travel to the right. The signal itself suffers a passing loss, a crossing loss, and a passing loss. It also generates a crossing crosstalk to up.

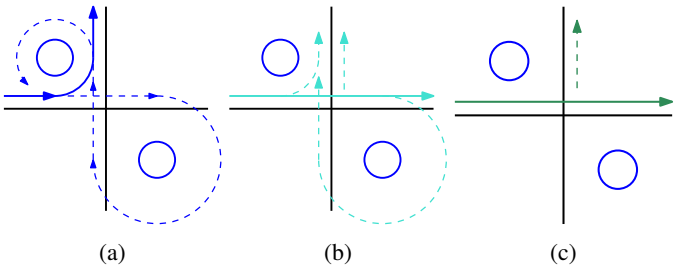
Signals coming from the down input port are skipped, as the process is rotationally symmetric to Fig. 19.

Fig. 20a shows a resonant crosstalk from the left input port drops by an MRR and goes to up. It suffers a drop loss.

Fig. 20b shows a nonresonant crosstalk from the left input port travels through a crossing to the right. The crosstalk suffers a passing loss, a crossing loss, and a passing loss. Crosstalks coming from the down input port is rotationally symmetric to Fig. 20 and is thus skipped.

Based on the analysis of 4 different CSE variations above, we are able to record all the signals and crosstalks travelling through every communication block. One step further, this algorithm also calculates the SNR for every signal. For example, in a 4×4 communication network like Fig. 12, using the value of signals and crosstalks in the output ports of block (0, 0), (0, 1), (0, 2), SNR for every signal can be calculated. Thus, this algorithm can find the signal suffering the worst-case SNR. The worst-case SNR determines the quality of signal transmission of an optical router. The optimization for the worst-case SNR is essential for WRONoCs.

The difficult point in crosstalk analysis is that new crosstalks are generated continuously and change directions according to resonant MRRs. This makes it hard to trace the path of all the crosstalks. The principle of this crosstalk analysis algorithm is that it does not trace the path of signals or crosstalks, yet it dynamically generates addresses to store the data of signals



(a) (b) (c)

Fig. 19: Signals from left. (a) Resonant signal from left drops to up and generates a resonant crosstalk which is dissipated. (b) Nearest nonresonant signals from left travel to right and generate a nonresonant crosstalk, a crossing crosstalk, and a nonresonant crosstalk to up. (c) Other nonresonant signals from left travel to right and generate a crossing crosstalk.

and crosstalks travelling out of each communication block, i.e. the energy value in dB and the wavelength of the signal or crosstalk. Thus, this algorithm is able to capture and analyze any energy transition inside a communication block. The path of a signal or a crosstalk in the router is not concerned and is generated automatically. This algorithm can be extended easily. More parameters like thermal effects can be integrated.

VI. Optimization Process And Experimental Results

In this section, the optimization process and experimental results are introduced. First, we disable crosstalk analysis and SNR optimization, and we focus only on optimizing MRR usage, wavelength usage, and the worst-case insertion loss. The corresponding experimental results are compared with Customtopo [13]. Second, we enable crosstalk analysis and SNR optimization. The optimization process is also changed in order to maximize the optimization for the worst-case SNR. Without sacrificing the optimized results of MRR usage, wavelength usage, and the worst-case insertion loss, FAST+ realizes a 1.75× optimization for the worst-case SNR.

A. The optimization for MRR usage, wavelength usage, and the worst-case insertion loss

1) Optimization process

Input: A communication network e.g. Fig. 2a and the insertion loss values [9] [13].

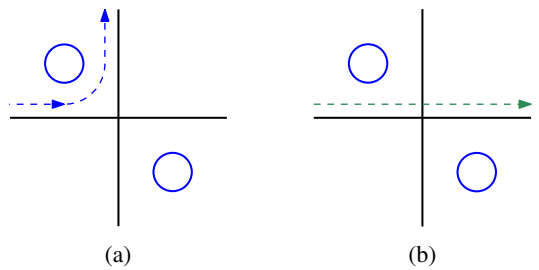
Output: Multiple optimized topologies with their communication dictionaries, MRR usage, the worst-case insertion loss, minimal wavelength usage, and wavelength assignment information.

Now, we introduce the optimization process step by step:

Step 1: Find redundant senders and receivers. Clear out all empty default paths in the topology as shown in Fig. 3b. Then we randomly order the remaining senders/receivers, make communication dictionaries for senders and receivers based on this port order. Then we generate an initial topology based on the communication dictionaries.

Next, reorder the senders and the receivers to generate new communication dictionaries and new topologies. This part of the algorithm is called **variation generator**. To reduce the number of variations and generate high quality variations, we propose three reduction techniques:

- Taking Fig. 3b as an example, we don't move or break empty default paths. To do this, we don't change the positions of S1 and R3. Empty default paths can be directly removed in physical layout. Once they are cleared out, they should not be broken.
- We don't move or break crossings with two MRRs. If this structure is broken down, the two MRRs could occupy two crossings. We want more empty crossings because they are removable in the physical layout.
- Fix the paths of all default communications. If default communications become communications requiring MRRs, extra MRRs have to be added in the topology.



(a) (b)

Fig. 20: Crosstalks from left. (a) A resonant crosstalk from left drops to up, suffering a drop loss. (b) A nonresonant crosstalk from left travels to right, suffering a passing loss, a crossing loss, and a passing loss.

With these reduction techniques, the search space is significantly reduced. Variations which are worse than the initial topology are ignored. Thousands of topologies which are at least as good as the initial topology can be generated within 1 second. In the code, we restrict the topology generation time to maximal 1 second. It is enough to find multiple topologies which are equivalent or better than the state of the art.

Step 2: Select the topologies which simultaneously have the minimal MRR usage, the smallest worst-case insertion loss, and the minimal N_{max} (indication of wavelength usage) among all generated topologies. After that, the topologies with the smallest number of non-empty crossings are again selected from these optimized topologies. Fewer non-empty crossings means more crossings with two MRRs and more default communications. This step finds the sparsest topologies, benefiting the physical layout.

Step 3: Run the ILP model to formally determine the minimal wavelength usage and assign a wavelength to each MRR for the chosen topologies. Finally, multiple optimized topologies (solutions) with their communication dictionaries, MRR usage, the worst-case insertion loss, minimal wavelength usage, and wavelength assignment information are printed as outputs. Here is an example of one of the solutions for a 4×4 network: (24) is the communication dictionary of the solution. Fig. 21a is the optimized topology in matrix form. Using (24) and Fig. 21a, it is easy to draw the topology in Fig. 21b.

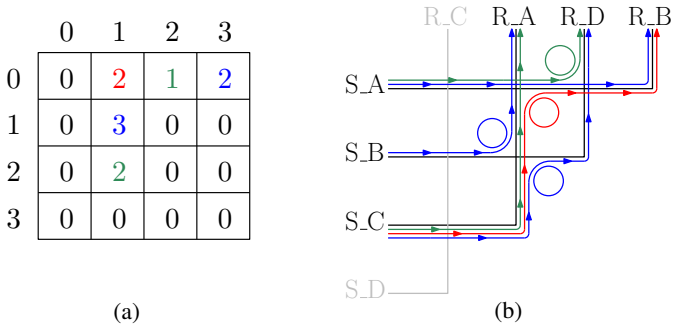
$$\begin{cases} sender\ dict.\text{:}\ (0\text{:}A,\ 1\text{:}B,\ 2\text{:}C,\ 3\text{:}D) \\ receiver\ dict.\text{:}\ (0\text{:}C,\ 1\text{:}A,\ 2\text{:}D,\ 3\text{:}B) \end{cases} \tag{24}$$

2) Experimental Results

We use Python to implement FAST+. The ILP model is solved by Gurobi [17], a mixed integer linear programming solver. To compare FAST+ and CustomTopo comprehensively, we test all 7 cases tested in CustomTopo. The results are shown in Table II from case 1 to case 7. To show that FAST+ can solve large communication networks, we test two 40×40 networks (case 8 and 9). Case 8 [18] is a sparse network with 32 communications. Case 9 is a synthetic case with 780 communications designed to test the speed of FAST+.

CustomTopo runs on a computer with dual Xeon processors under 2.67GHz base frequency in C++ [13]. FAST+ runs on a Core i5 single processor computer under 1.6GHz base frequency in Python. Despite running on a much weaker computer, FAST+ is still much faster and provides multiple competitive topologies.

- FAST+ uses 13% fewer MRRs and 7% fewer wavelengths than CustomTopo on average. For the worst-case insertion

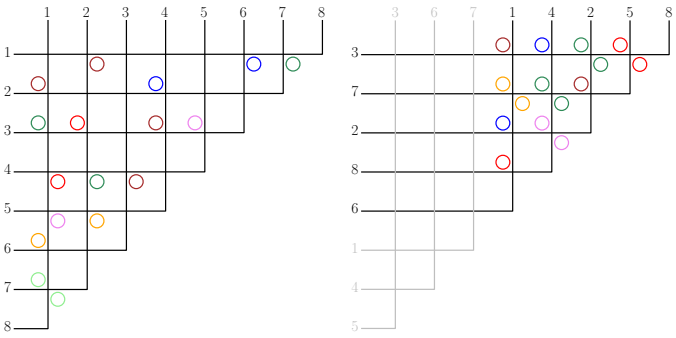


(a) (b)

Fig. 21: (a) The optimized topology in matrix form. This is the original output of FAST+. Different colors represent different wavelengths assigned to each MRR. The wavelength assignment and usage optimization are done using the ILP model introduced in subsection IV-B. (b) The final topology drawn through combining (a) and the communication dictionary (24). S_A means sender A, R_A means receiver A. The gray path is an empty default path cleared out by FAST+. Solid lines with arrows are the optical signals.

loss, FAST+ and CustomTopo match each other move for move.

- For case 1 to 7, FAST+ is 1007 times faster and can find 4 times more optimized solutions on average. This makes FAST+ thousands times faster than CustomTopo while providing better solutions.
- For case 1, 5, 7, and 9, FAST+ outputs one optimized solution for each case due to the reduction techniques. No variations are generated because FAST+ considers that the initial topology is already good enough.
- Most importantly, FAST+ can easily solve communication networks with large sizes, such as 40×40 or more (case 8 and 9). CustomTopo is too slow to solve networks with such a size. Moreover, for case 8, FAST+ clears out 13 empty default paths. 13 senders and 13 receivers can be directly removed. By doing this, not only 26 physical ports but also 429 empty crossings are saved. This gives FAST+ a significant advantage in the physical layout. This feature is illustrated in Fig. 22 with an 8×8 network example. Fig. 22a and Fig. 22b support the same connection. In Fig. 22b: three senders, three receivers, and 18 empty crossings are directly eliminated.



(a) (b)

Fig. 22: (a) Initial topology. (b) Clearing out three empty default paths based on (a).

B. The Optimization for SNR

A good WRONoC topology should support the demanded connectivity with low MRR usage and low wavelength usage. Moreover, optical signals should have low insertion loss and high SNR after travelling through the router. In Subsection VI-A, we optimized MRR usage, wavelength usage, and the worst-case insertion loss. In comparison with CustomTopo, FAST+ is much more efficient and provides competitive solutions. Now, we try to optimize the worst-case SNR for the same test cases without sacrificing MRR usage, wavelength usage, and the worst-case insertion loss.

1) Optimization process

Input: A communication network; the insertion loss and crosstalk values [9] [13] [21].

Output: Multiple optimized topologies with their communication dictionaries, MRR usage, the worst-case insertion loss, minimal wavelength usage, wavelength assignment information, and SNR values for all the signals.

Step 1: Clear out empty default paths and generate topology variations. For sparse networks (case 2, 3, 4, 6, 7, 8, 9), we generate variations for ten seconds, so that enough variations can be generated. For dense networks (case 1, 5), we disable the three reduction techniques and generate variations for three seconds, so that dense networks also have numerous variations. Ten seconds and three seconds are determined to balance the results and the runtime. If we generate variations for longer time, better variations are likely to be generated.

Step 2: Select the topologies which simultaneously have the minimal MRR usage, the smallest worst-case insertion loss, and the smallest N_{max} among all generated topologies. Then again select the variations with the smallest number of non-empty crossings from the selected topologies.

Step 3: Run the ILP model to formally determine the

TABLE II: Comparison between CustomTopo and FAST+

Idx	d	N	Method	MRR	W	$\mathbf{I_{worst}}$	V	Time
1	8	44	CustomTopo	48	7	0.85	1	53s
			FAST+	36	7	0.835	1	0.04s
2	12	26	CustomTopo	26	8	0.8	1	184s
			FAST+	24	7	0.77	4	1.25s
3	12	20	CustomTopo	18	5	0.6	1	14s
			FAST+	14	5	0.64	7	1.65s
4	16	22	CustomTopo	20	7	0.7	1	13s
			FAST+	19	7	0.73	5	1.50s
5	8	48	CustomTopo	40	6	0.9	1	138s
			FAST+	40	6	0.9	1	0.04s
6	8	24	CustomTopo	24	7	0.8	1	3s
			FAST+	20	6	0.82	10	0.31s
7	8	24	CustomTopo	24	7	0.8	1	63s
			FAST+	24	6	0.8	1	0.03s
8	40	32	FAST+	31	3	0.635	1	1.32s
9	40	780	FAST+	741	20	2.3	1	53.5s

Idx: index of test cases; **d**: $degree$ (size of communication matrix, 8 means 8×8 communication matrix.); **N**: total number of communications in the network; **MRR**: total number of MRRs; **W**: total number of wavelengths. In CustomTopo [13], only wavelengths assigned to ADFs are counted, one more wavelength has to be added for default communications.; $\mathbf{I_{worst}}$: the worst-case insertion loss in dB, crossing losses caused by empty crossings are not counted [13] ; **V**: number of variations; **Time**: the program runtime in seconds.

minimal wavelength usage and assign a wavelength to each MRR for the chosen topologies. SNR analysis requires the wavelength assignment information.

Step 4: Select the variations whose worst-case SNR are the highest among selected variations. By doing this, we optimize SNR while not sacrificing MRR usage, wavelength usage, and the worst-case insertion loss. Actually they are at least as good as Table II because many more variations are generated.

2) Experimental Results

We analyze and optimize the worst-case SNR for all 9 cases tested in Table II. The optimized results are shown in Table III (SNR1). The results can be better if more variations are generated using the variation generator of FAST+.

To check how good the optimized results are, we use the variation generator in FAST+ to generate random topology variations for 0.2 seconds for case 1 to 7. Three reduction techniques are disabled. No selection for MRR usage, wavelength usage, and the worst-case insertion loss is performed. SNR is calculated for all generated variations. We use the lowest worst-case SNR among these variations (shown in Table III (SNR0)) to compare with the optimized results (SNR1). For an 8×8 communication network, only about 390 variations can be generated in 0.2 seconds. But totally there are $(8!)^2$ variations. So SNR0 can be worse in reality if the sender/receiver orders of the communication network are random. For case 8 and 9, 0.2 seconds are not long enough to generate variations that make differences in the worst-case SNR, because the sizes of these cases are too large. For case 8 and 9, we manually reorder the senders and receivers, to quickly change the sequences of the senders/receivers globally. Results show that the optimized worst-case SNR is 1.75× better than the worst-case SNR without optimization on average. The optimization varies a lot for different cases because the initial sender/receiver orders of some cases already offer good SNR.

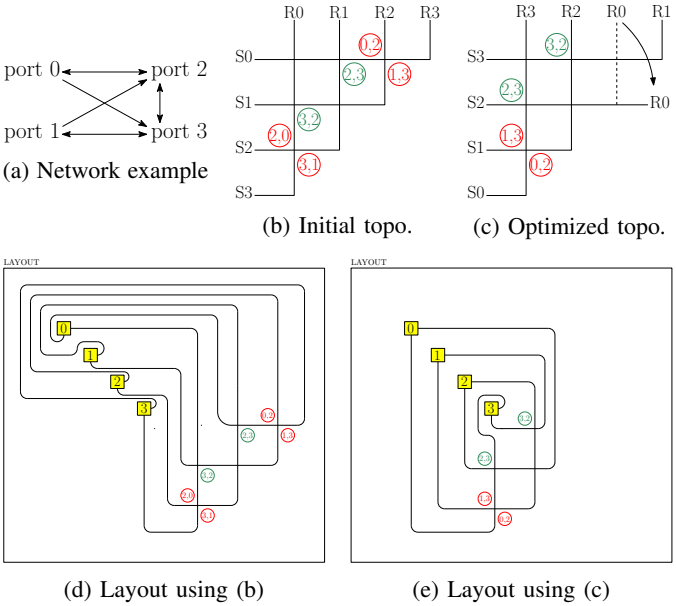
VII. Discussion: Physical Layout

FAST+ optimizes the topology by initiatively changing the sequences of senders and receivers. This feature gives FAST+ a direct connection with the physical layout, making it perform even better when considering physical constraints, e.g. physical port locations. In this section, we introduce three

TABLE III: SNR Optimization

Idx	d	N	SNR0	SNR1	Imp.	V	t
1	8	44	43.3	62	1.43×	24	18.4s
2	12	26	34.9	45.9	1.32×	8	15.5s
3	12	20	71	125.8	1.77×	2	24.2s
4	16	22	53	53	1×	1	20.5s
5	8	48	63.2	127.9	2.02×	16	3.6s
6	8	24	56.6	58.9	1.04×	250	13.3s
7	8	24	50.3	234.1	4.65×	1	0.04s
8	40	32	95.6	121.1	1.27×	4	14.1s
9	40	780	6.9	8.8	1.28×	1	52.4s

Idx: index of test cases; **d**: $degree$ (size of communication matrix, 8 means 8×8 communication matrix.); **N**: total number of communications in the network; **SNR0**: the worst-case SNR without optimization. For case 1, "SNR0 = 43.2" means the signal power is 43.2 times bigger than the noise power.; **SNR1**: the optimized worst-case SNR; **Imp.**: improvement; **V**: number of variations; **t**: the program runtime in seconds.



(a) Network example (b) Initial topo. (c) Optimized topo.

(d) Layout using (b) (e) Layout using (c)

Fig. 23: Physical layout comparison. (b) The initial topology for (a). (c) One of the optimized topologies for (a).

exclusive advantages in FAST+ which help us draw better layout.

First, FAST+ can find redundant senders/receivers, adjust the order to create empty default paths and clear out empty default paths (Fig. 22). This function directly saves waveguides, crossings, and physical ports in the physical layout.

Second, in the physical layout, the different sender/receiver orders in FAST+ help eliminate empty crossings inside the topology while not adding crossings outside the topology. Fig. 23c shows an optimized topology with different sender/receiver orders for Fig. 23a. Sender order is S3, S2, S1, S0, receiver order is R3, R2, R0, R1 (note the order difference between Fig. 23b and Fig. 23c). This difference enables the elimination of empty crossings both inside and outside the topology in the physical layout.

Third, most importantly, the state of the art ignores the physical position of communication ports. In the physical layout, if the orders of senders/receivers in topology do not match with the physical port locations, waveguide detours have to be introduced. For example, the layouts in Fig. 23d and Fig. 23e have equivalent chip areas and physical port locations. Fig. 23e has significantly shorter waveguides due to matched port orders. In FAST+, the sequences of senders and receivers are changeable. Given the physical information, FAST+ can generate topologies with matched port orders. This is especially beneficial to dense networks as different port orders won't worsen the topology but will significantly improve the layout. In Table. II, FAST+ only generates one solution for case 1, 5, 7. This is because more variations won't improve the topology for these cases. But in the physical layout, the reduction techniques can be disabled and many variations can be generated for every case. For sparse networks, FAST+ always provides multiple topology variations for the physical layout to find the best tradeoff between topology and layout.

Based on the half-matrix topology, FAST+ proposes empty default paths, different sender/receiver orders and optimized

variations to eliminate empty crossings and avoid waveguide detours. These three features link FAST+ directly to the physical layout and make FAST+ not only an efficient topology customization algorithm, but also a promising layout platform.

VIII. Conclusion

In this work, we propose a crosstalk-aware WRONoC topology customization and optimization tool for application-specific designs. This paper introduces a concrete router-level crosstalk analysis tool and realizes SNR optimization. When SNR is not considered, the combination of an ILP model and a special sweeping technique makes the algorithm (called FAST+) thousands times faster than the state of the art while providing multiple better solutions. When SNR is analyzed and optimized, FAST+ provides optimized SNR which is 1.75× better than the initial values on average while not sacrificing MRR usage, wavelength usage, and the worst-case insertion loss. Moreover, empty default paths, different sender/receiver orders, and many optimized variations help FAST+ eliminate redundant senders/receivers, empty crossings, and waveguide detours in the physical layout. These three features give FAST+ a natural connection with the physical layout, making it not only an efficient topology customization algorithm but also a promising layout platform. We aim at developing a competitive automatic physical layout tool in the future work.

References

[1] T.-M. Tseng et al., "Wavelength-Routed Optical NoCs: Design and EDA - State of the Art and Future Directions," *Int. Conf. Comput.-Aided Des.*, pp. 1-6, 2019.

[2] A. Truppel et al., "PSION+: Combining Logical Topology and Physical Layout Optimization for Wavelength-Routed ONoCs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, Iss. 12, pp. 5197-5210, 2020.

[3] S. L. Beux et al., "Reduction Methods for Adapting Optical Network on Chip Topologies to 3D Architectures," *Microprocessors and Microsystems: Embedded Hardware Design*, vol. 37, no. 1, pp. 8798, 2013.

[4] Y. Xie et al., "Crosstalk Noise and Bit Error Rate Analysis for Optical Network-on-Chip," *Proceedings of the 47th Design Automation Conference*, 2010.

[5] M. Nikdast et al., "Fat-Tree-Based Optical Interconnection Networks Under Crosstalk Noise Constraint," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 1, pp. 156-169, 2014.

[6] M. Tala et al., "Populating and Exploring the Design Space of Wavelength-Routed Optical Network-on-Chip Topologies by Leveraging the Add-Drop Filtering Primitive," *IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*, pp. 18, 2016.

[7] M. Li et al., "Maximizing the Communication Parallelism for Wavelength-Routed Optical Networks-On-Chips," *25th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 109-114, 2020.

[8] W. Bogaerts et al., "Silicon Microring Resonators," *Laser Photonics Rev.*, pp. 47-73, 2012.

[9] M. Nikdast et al., "Crosstalk Noise in WDM-Based Optical Networks-on-Chip: A Formal Study and Comparison," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, pp. 2552-2565, 2015.

[10] L. Ramini et al., "Contrasting Wavelength-Routed Optical NoC Topologies for Power-Efficient 3D-stacked Multicore Processors using Physical-Layer Analysis," *Proc. Design, Automation, and Test Europe Conf.*, pp. 1589-1594, 2013.

[11] M. Brière et al., "System Level Assessment of an Optical NoC in an MPSoC Platform," *Proc. Design, Automation, and Test Europe Conf.*, pp. 1084-1089, 2007.

[12] X. Tan et al., "On a Scalable, Non-Blocking Optical Router for Photonic Networks-on-Chip Designs," *Symp. Photonics and Optoelectronics (SOPO)*, 2011.

[13] M. Li et al., "CustomTopo: A Topology Generation Method for. Application-Specific Wavelength-Routed Optical NoCs," *Int. Conf. Comput.-Aided Des.*, 2018.

[14] M. Xiao et al., "FAST: A Fast Automatic Sweeping Topology Customization Method for Application-Specific Wavelength-Routed Optical NoCs," *Proc. Design, Automation, and Test Europe Conf.*, pp. 1651-1656, 2021.

[15] S. L. Beux et al., "Optical Ring Network-on-Chip (ORNoC): Architecture and Design Methodology," *Design, Automation, and Test Europe Conf.*, pp. 788-793, 2011.

[16] D. König, "Über Graphen und ihre Anwendung auf Determinantentheorie und Mengenlehre," *Mathematische Annalen*, vol. 77, no. 4, pp. 453465, 1916.

[17] Gurobi Optimization, Inc., Gurobi Optimizer Reference Manual. http://www.gurobi.com.

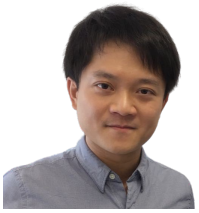
[18] J. Hu et al., "System-Level Buffer Allocation for Application-Specific Networks-on-Chip Router Design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 12, pp. 2919-2933, Dec. 2006.

[19] L. H. K. Duong et al., "Coherent and Incoherent Crosstalk Noise Analyses in Interchip/Intrachip Optical Interconnection Networks," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, pp. 2475–2487, 2016.

[20] J. Chan et al., "Architectural Exploration of Chip-Scale Photonic Interconnection Network Designs Using Physical-Layer Analysis," *J. Lightwave Technol.*, pp. 1305–1315, 2010.

[21] K. Preston et al., "Performance guidelines for WDM interconnects based on silicon microring resonators," *IEEE, Laser Science to Photonic Applications*, 2011.

Moyuan Xiao received the bachelor degree in electrical engineering and information technology from University of Duisburg-Essen (Uni-DUE), Duisburg, Germany, in 2018, the master degree in electrical engineering and information technology from Technical University of Munich (TUM), Munich, Germany, in 2020. His research interests lay on efficient computer-aided design and optimization for emerging technologies, such as optical networks-on-chips. Since 2018, he has been developing reliable hardware and software for self-driving cars, such as emergency brake system, vehicle domain controller, automotive Ethernet, and high-performance driver assistance algorithms for SoCs with low computing capability. He won the championship of Formula Student Germany, Austria, and Spain in 2019, representing TUM's Tufast racing team.



Tsun-Ming Tseng (S'10–M'15) received the bachelor degree in electronics engineering from National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 2010, the M.Sc. and the Dr.-Ing. degrees from Technical University of Munich (TUM), Munich, Germany, in 2013 and 2017, respectively. He leads a research group in the Chair of Electronic Design Automation, TUM. His research interests focus on design automation for emerging technologies, such as microfluidic biochips and optical networks-on-chips.

Ulf Schlichtmann (S'88–M'90–SM'18) received the Dipl.Ing. and Dr.-Ing. degrees in electrical engineering and information technology from Technical University of Munich (TUM), Munich, Germany, in 1990 and 1995, respectively. He was with Siemens AG, Munich, and Infineon Technologies AG, Munich, from 1994 to 2003, where he held various technical and management positions in design automation, design libraries, IP reuse, and product development. He has been a Professor and the Head of the Chair of Electronic Design Automation, TUM, since 2003, where he also served as Dean of the Department of Electrical and Computer Engineering, from 2008 to 2011, and as Associate Dean of Studies of International Studies since 2013. Since 2016, he is an elected member of TUM's Academic Senate and Board of Trustees. He is a member of the German National Academy of Science and Engineering and also serves on a number of advisory boards. Ulf's current research interests include computer-aided design of electronic circuits and systems, with an emphasis on designing reliable and robust systems. In recent years, he has increasingly worked on emerging technologies, such as microfluidic biochips and optical interconnect.