Technical University of Munich
TUM School of Computation Information and Technology
Chair of Electronic Design Automation

Bandwidth-Aware Topology Synthesis for Wavelength-Routed Optical Networks-on-Chips

Master's Thesis

Fares Abid

Technical University of Munich
TUM School of Computation Information and Technology
Chair of Electronic Design Automation

Bandwidth-Aware Topology Synthesis for Wavelength-Routed Optical Networks-on-Chips

Master's Thesis

Fares Abid

Advisor : Mengchu Li, M.Sc.
Advising Professor : Prof. Dr.-Ing. Ulf Schlichtmann
Topic issued : 03.08.2023
Date of submission : 01.02.2024

Fares Abid

Acknowledgement

Dedicated to my loving parents, Lobna & Hamadi, for the care and continuous support they have given me throughout my life.

I would like to thank my brothers and friends for their support.

I would also like to thank my advisor, Ms Mengchu Li, for the support and the valuable feedback.

Abstract

Wavelength-routed optical networks-on-chips (WRONoCs) is a next generation multiprocessor system-on-chip architecture for communication among processors. A recent research article demonstrates that communication parallelism can be improved by using different wavelengths in each path of a given WRONoC topology. This mathematical modeling method, which only works for a given topology, assigns one radius to each silicon microring resonators (MRR) type to improve communication parallelism. The effectiveness of this method depends on the given topology. To further improve communication parallelism, this paper presents a systematic approach which generates a topology that considers communication parallelism in topology design. The proposed topology aims to maximize the worst case parallelism and the aggregate parallelism for full connectivity as well as for customized connectivity. The performance of the proposed topology is compared with the performance of several topologies, such as: general purpose topologies, CustomTopo, the Light topology and the state-of-the-art parallelism-aware topology that aims to maximize the worst case parallelism. Experimental results show that the proposed topology outperforms all previous works in terms of improving the worst case parallelism and the aggregate parallelism.

Keywords: Bandwidth-Aware Topology, Communication Parallelism, Wavelength-routed optical networks-on-chips.

Contents

List of Figures

1. Introduction

The on chip communication in multiprocessor system-on-chip (MPSoC) has increased greatly in the past decade. To deal with this huge increase, optical networks-on chips (ONoCs) are proposed (Vantrease et al., 2008). These optical networks on chips adopt optical connections to transmit data instead of electrical connections. Thus, ONoCs provide ultra low signal delay as well as high bandwidth (Yan et al., 2020). There are two main classes of ONoCs based on the routing mechanism (Werner, Navaridas, & Luján, 2017):

- Control-networks-based: In this category, only one sender and one receiver can communicate at any given time in order to avoid data collision. In order to send a data from a sender (master) to a receiver (slave), an additional control network must reserve a signal path connecting the slave and the master (Koohi & Hessabi, 2012) (Parini, Ramini, Bellanca, & Bertozzi, 2011).

- Wavelength-routed: Wavelength-routed optical networks-on-chips (WRONoCs) route each optical signal according to its wavelength using silicon microring resonators (MRRs). In fact, WRONoCs setup signal paths at the design stage, so that all masters can communicate with all their slaves at the same time. Thus, WRONoCs do not contain energy and latency overhead for arbitration in contrast to other ONoC architecture (Truppel, Tseng, Bertozzi, Alves, & Schlichtmann, 2019) (Manolatou & Haus, 2002).

Most previous research about WRONoCs assumes that each sender transmits one signal at a time to a receiver. However, an MRR can support different wavelengths. Using MRRs to route optical signals according to their wavelengths, several signals of different wavelengths can be transmitted concurrently between the same sender-receiver pair. Taking advantage of this fact, Li *et al.* (2020) proposed a mathematical approach, in which several wavelengths are assigned to each path, resulting in an increase in the communication parallelism of a given topology (Li, Tseng, Tala, & Schlichtmann, 2020). There exist several topologies:

- General purpose topologies: These topologies support full connectivity, i.e., all senders are connected to all receivers. The most common topologies are λ-router (Briere et al., 2007) and Snake (Ramini, Grani, Bartolini, & Bertozzi, 2013a). There are other topologies where each sender can communicate with all receivers other than its corresponding receiver (the receiver with the same id).

- Application-specific topologies: These topologies are designed for cases where specific senders are required to communicate with specific receivers. This information is given in the communication matrix (Li, Tseng, Bertozzi, Tala, & Schlichtmann, 2018). The application-specific topology can be constructed by removing the redundant MRRs in a general purpose topology.

Both general purpose topologies and the existing application-specific topologies do not take into account communication parallelism in topology design. The approach in (Li et al., 2020) depends on the given topology which may not suit the communication matrix, resulting in only a few wavelengths being assigned to each path. The approach in (Chen et al., 2023) improves the worst case parallelism in topology design. However, it does not aim to maximize the aggregate parallelism. Therefore, this master's thesis presents a systematic approach that considers communication parallelism in topology design in order to improve the worst case parallelism and the aggregate parallelism.

The remainder of this report is organized as follows: In the second chapter, optical networks on chip and their components are presented. Later, in the third chapter the mathematical modeling methods that aim to maximize communication parallelism are described. Next, the bandwidth-aware topology synthesis for wavelength-routed optical networks-on-chips is presented. Subsequently, in the fifth chapter, the results of the benchmark and their interpretations are discussed. Finally, the last chapter summarizes the project and concludes the master's thesis.

2. Optical Networks-on-Chip

Optical networks-on-chips (ONoCs) is a new interconnection technology which is used to connect the processors on a chip (Yan et al., 2020). ONoCs adopt optical connections to connect the processors, providing high bandwidth, low delay and low power consumption (Yan et al., 2020). Additionally, ONoCs have great potential in high-speed communication networks (Yan et al., 2020). The first section presents the main components of ONoCs. Then, the second section presents WRONoC communication parallelism. Afterwards, the third section presents the power model of the main components of ONoCs.

2.1. The Main Components of Optical Networks-on-Chips

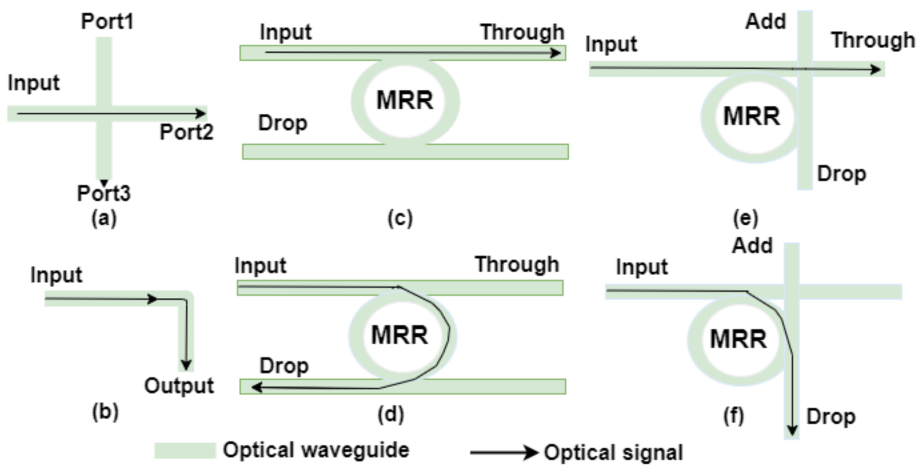

Figure 2.1.: The basic ONoCs components (Yan et al., 2020).

There are several basic ONoCs components, which include microring resonators, waveguides, waveguide bends and waveguide crossings (Yan et al., 2020). Figure 2.1 presents the most commonly used basic ONoCs components (Yan et al., 2020): (a) Waveguide crossing, (b) Waveguide bend, (c) Parallel switch element (PSE) (State: OFF), (d) PSE (State: ON), (e) Cross switch element (CSE) (State: OFF) , (f) CSE (State: ON).

The key element that enables signals to be switched among different waveguides is the silicon microring resonator (MRR). An MRR consists of a microring waveguide, which is looped. Each MRR has a radius. Each radius value corresponds to different resonance wavelengths. For an MRR of radius r, Equation 2.1 is used to calculate the corresponding resonance wavelengths $\lambda_{r,i}$, where the constant n_{eff} represents the refractive index (Bogaerts et al., 2012).

$$\lambda_{r,i} = \frac{2\pi r \times n_{eff}}{i}, \qquad i \in \{1, 2, 3, ..., \} \tag{2.1}$$

The signal will be coupled to the ring of the MRR m and there will be a direction change of the signal (The signal ***activates*** the MRR m), if and only if the wavelength of a signal matches one of the resonance wavelengths of m. In this case, the signal is ***on-resonance with m***. Otherwise, if the wavelength of the signal does not match any of the resonance wavelengths of the MRR, the signal ignores the MRR. The signal will not be coupled to the ring, and will continue to propagate along the original waveguide. In this case, the signal is ***off-resonance with m***.

The MRRs are used to construct the CSEs and PSEs. The MRR in the CSE and the PSE enables the routing mechanism (Yan et al., 2020). In fact, the CSE consists of at least one MRR, and two crossed optical waveguides. The PSE consists of a microring resonator (MRR) and two parallel waveguides. As mentioned before, depending on the signal wavelength and the radius of the MRR, a signal can be either on-resonance with the MRR of the CSE/PSE or off-resonance with the MRR of the CSE/PSE.

- The PSE: if the signal is on-resonance with the MRR of the PSE, the optical signal is first coupled to the ring and then to the waveguide parallel to the original waveguide, leading to a 180-degree direction change (Yan et al., 2020). Otherwise, if the signal is off-resonance with the MRR of the PSE, the signal ignores the MRR and continues to propagate along the original waveguide (see Figure 2.1(c)) (Yan et al., 2020).

- The CSE: if the signal is on-resonance with the MRR of the PSE, the optical signal is coupled to the ring and then to another waveguide, leading to a direction change depending on the input port (see Figure 2.1(f)) (Yan et al., 2020). If the input port is the Add port, the direction change is 270 degrees. If the input port is port 1, the direction change is 90 degrees. The power loss and crosstalk noise caused by the 270-degree direction change is very large. Therefore, most designers only focus on using port 1 (Zhang et al., 2014) (Lin & Lea, 2012). Otherwise, if the signal is off-resonance with the MRR of the CSE, the signal ignores the MRR and continues to propagate along the original waveguide (see Figure 2.1(e)) (Yan et al., 2020).

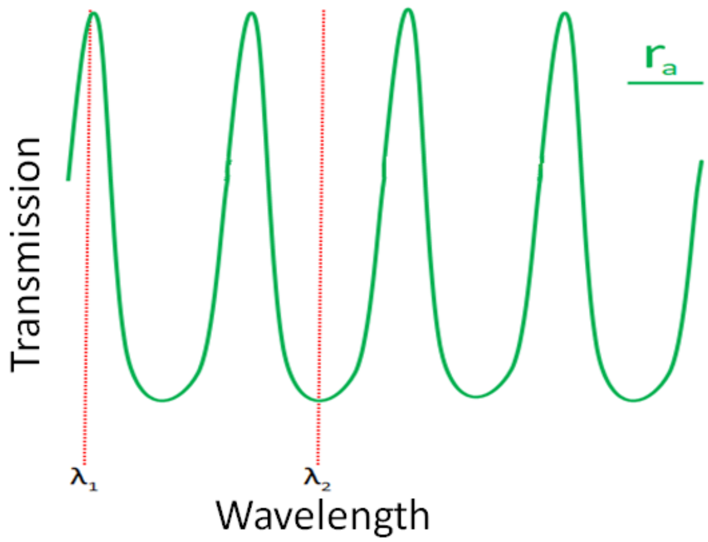

Figure 2.2.: The transmission optical spectrum of MRR a.

The behaviour of the wavelength with respect to the MRR are explained using the following examples: The periodic transmission spectrum of an MRR is presented in Figure 2.2. The spectrum has four peaks, which represent four resonant wavelengths. λ_1 matches one peak of the spectrum of the MRR. Therefore, when a signal on wavelength λ_1 goes through the MRR of CSE (from the input port), the direction of the signal will be changed by 90 degrees. On the other hand, λ_2 does not match on peak of the spectrum of the MRR. Therefore, a signal on wavelength λ_2 ignores the MRR, therefore, the signal is transmitted to the through port. An MRR can support several resonant wavelengths. These MRRs allow each master to communicate with each of its slaves using many non-overlapping wavelengths simultaneously (Li et al., 2020).

2.2. WRONoC Topologies

2.2.1. The Working Mechanism of WRONoCs

The WRONoC working mechanism is explained using the following example: Figure 2.3 presents a simple WRONoC topology. In this example, the wavelength λ_i is only on-resonance with the MRR m_i. As can be seen, with 4 different wavelengths (λ_1, λ_2, λ_3 and λ_4), master 1 can send data to its four slaves at the same time. The wavelengths λ_1, λ_2 and λ_3 are coupled to the blue MRR, the green MRR and the violet MRR, respectively. λ_4 ignores the three MRRs and reaches slave 4. The path between master 1 and slave 4 is called default path because it does not resonate with any MRR in the path.

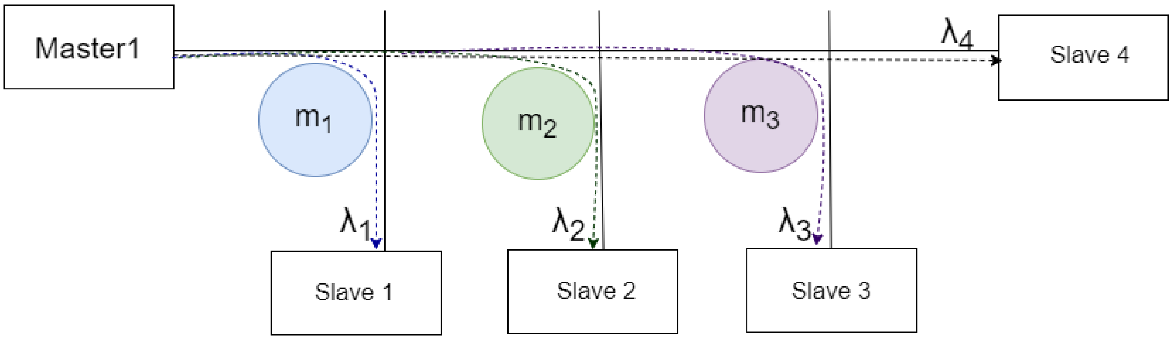

Figure 2.3.: A WRONoC topology.

In WRONoCs, it is possible to transmit all signals concurrently along all waveguides. Figure 2.4 presents 4x4 λ-router topology (Briere et al., 2007). All masters (I_1, I_2, I_3 and I_4) can communicate simultaneously with all slaves (T_1, T_2, T_3 and T_4). For example, I_1 can send data to T_1, T_2, T_3 and T_4 at the same time using 4 different wavelengths.

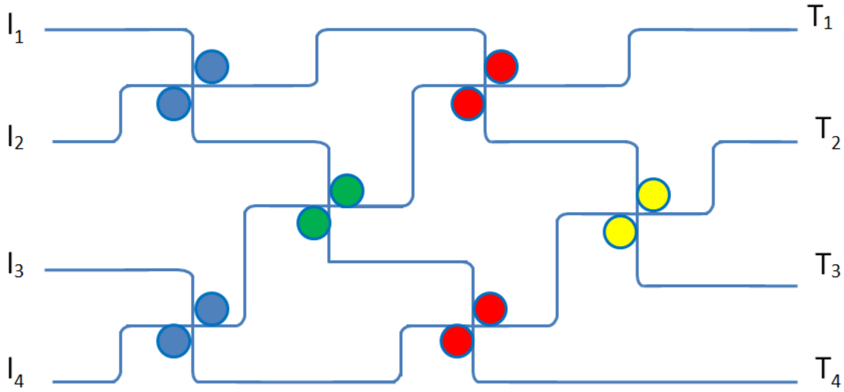

Figure 2.4.: The 4x4 λ-router topology (Briere et al., 2007).

2.2.2. Full Connectivity and Customized Connectivity

Several studies assume that in WRONoC topologies the communication graph is complete and that there is a connection between each master-slave pair (Briere et al., 2007) (Le Beux et al., 2011) (Peano, Ramini, Gavanelli, Nonato, & Bertozzi, 2016). However, in reality there exist several cases where a full connectivity is not required. For example, in processor-memory networks, one memory usually does not communicate with other memories (Wang et al., 2017). Therefore, using fully connected topologies for applications that do not need full connectivity can result in a large waste of area and resources (Li et al., 2018). Therefore, several topologies are designed based on the communication matrix to reduce the power loss as well as the MRR usage. Figure 2.5 presents the Customtopo topology, which aims to reduce the add-drop filter (ADF) and wavelength usage by designing only the paths that connect each sender to its receivers.

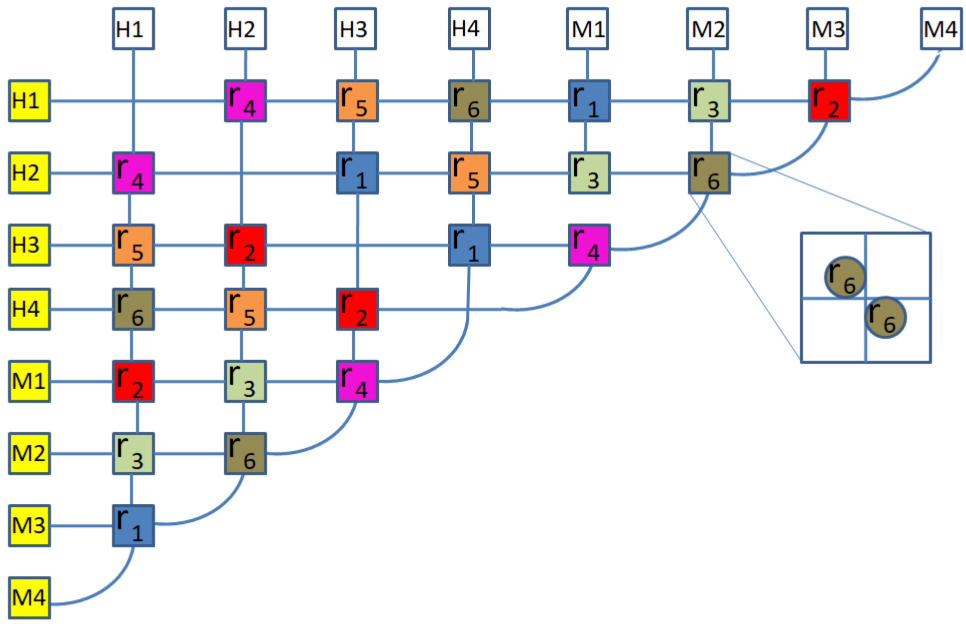

Figure 2.5.: A topology generated using CustomTopo

2.2.3. The Physical Layout

Several state-of-the-art topologies, such as λ-router, consider the senders and receivers as independent nodes and do not focus on putting each sender s_i close to the receiver with the same id r_i (Ramini et al., 2013a) (Briere et al., 2007). As can be seen in the λ-router topology in Figure 2.4, the senders are placed on the left side, while the receivers are placed on the right side. However, in actual WRONoC topologies, IP-core consists of a sender and a receiver. Therefore, the sender and the receiver of an IP-core should be placed close to each other. For example, core 1 consists of sender 1 and receiver 1. Thus, sender 1 should be placed close to receiver 1. Implementing the λ-router topology on a physical plane may cause many additional waveguide crossings by bringing each master close to its corresponding slave. For example, there are only 28 waveguide crossings in the 8x8 λ-router logic scheme, while in the λ router physical scheme there are 64 waveguide crossings (Ramini, Bertozzi, & Carloni, 2012). These additional waveguide crossings increase the power loss as well as the crosstalk noise. Figure 2.6 presents the 4x3 Light topology (Zheng et al., 2021). As can be seen, the design of this topology matches the physical layout constraint. In this master's thesis, the proposed topology considers the physical layout constraint and puts each master close to its corresponding slave.

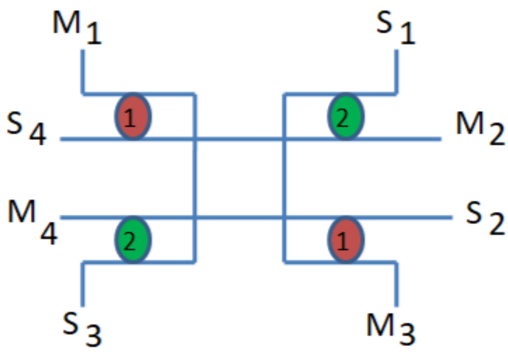

Figure 2.6.: The 4x3 Light topology

2.3. WRONoC Communication Parallelism

Different wavelengths can activate an MRR, as mentioned in Section 2.1. Thus, a sender can transmit several signals of different wavelengths concurrently to a receiver. The communication parallelism is explained using the following example: The 4*3 GWOR topology is presented in Figure 2.7 (Tan, Yang, Zhang, Jiang, & Yang, 2011). In this example, the MRRs have two different radii. The MRR in the same colour have the same radius. Figure 2.8 presents the optical transmission spectra of these two different MRRs. In this topology, the senders are I_0, I_1, I_2 and I_3 and the receivers are T_0, T_1, T_2 and T_3. Master I_0 can communicate with the slaves T_1 and T_2 using wavelengths that satisfy the following constraints:

- To send data to T_1, the wavelength should be on-resonance with MRR_a and off-resonance with MRR_b. The wavelength should not conflict with a resonance wavelength of MRR_b.

- To send data to T_2, the wavelength should be on-resonance with MRR_b.

λ_1, λ_4 and λ_6 are on-resonance with MRR_b. Therefore, λ_1, λ_4 and λ_6 can be used to send data from I_0 to T_2. λ_2, λ_5, λ_3 and λ_6 are on-resonance with MRR_a. It is not possible to use λ_6 to send data from master I_0 to slave T_1, because the optical signal is transmitted first through the MRR_b. Since the optical signal on wavelength λ_6 activates MRR_b, the direction of the signal will be changed and in this case the signal will reach T_2 instead of T_1.

A wavelength w_1 conflicts with another wavelength w_2 when the absolute difference between the two wavelengths (w_1 and w_2) is smaller than the safe distance δ (Li et al., 2020). A radius r conflicts with the wavelength w, if at least one resonant wavelengths supported by the radius r conflicts with w (Li et al., 2020). In other words, a radius r does not conflict with wavelength w if the constraint in Equation 2.2 is satisfied.

$$|\mathbf{w} - \lambda_{r,i}| > \quad \delta \qquad\qquad \forall \quad i \in \{1, ..., n\} \tag{2.2}$$

In the example in Figure 2.8, the safe distance is $\delta = 0.8$ nm. The absolute difference between λ_1 and λ_2 is 0.6 nm which is smaller than δ. Thus, λ_1 conflicts with λ_2; therefore, the radius $r = a$ conflicts with λ_1 and the radius $r = b$ conflicts with λ_2. As a result, λ_2 can not be used to send data from I_0 to T_1. λ_3 and λ_5 are on-resonance with MRR_a and off-resonance with MRR_b and do not conflict with MRR_b. Therefore, only λ_3 and λ_5 can be used to send data from I_0 to T_1.

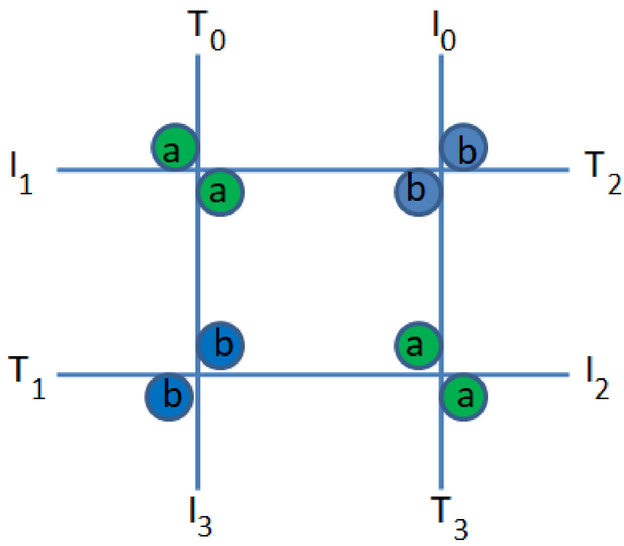

Figure 2.7.: The 4x3 GWOR topology.

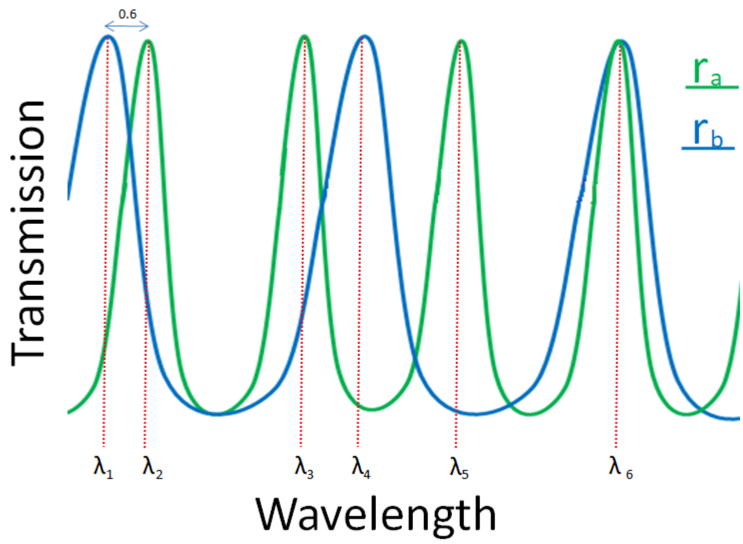

Figure 2.8.: The optical transmission spectra of two different MRRs: a and b.

2.3.1. Parallelism Calculation

There are two main terms regarding parallelism calculation:

- ***The aggregate parallelism***: The sum of the number of wavelengths used for each path in the topology, as can be seen in Equation 2.3, where P is the set of all signal paths in the topology and the function v(p) returns the number of wavelengths assigned to path p.

$$v_{aggregate} = \sum_{p \in P} v(p) \tag{2.3}$$

- ***The worst case parallelism***: The number of wavelengths of the path with the minimum number of wavelengths in the topology, as can be seen in Equation 2.4.

$$v_{worst} = \min_{p \in P} v(p) \tag{2.4}$$

2.4. Loss Model and Crosstalk Model

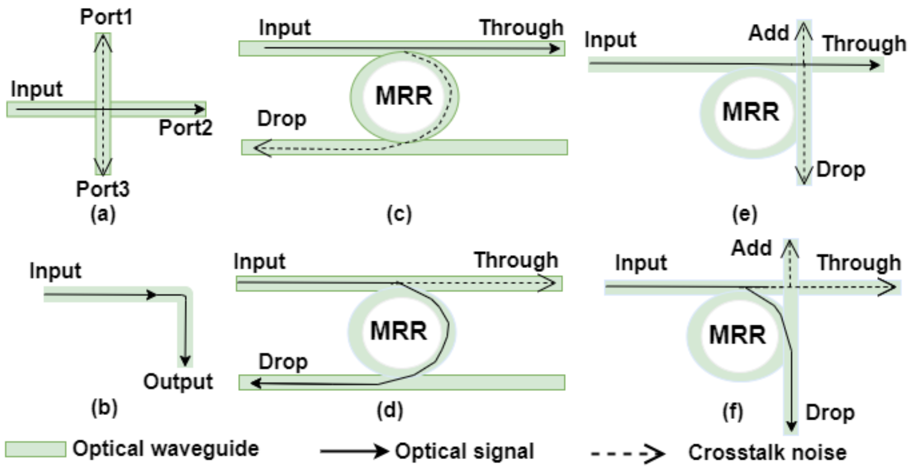

Figure 2.9.: The basic ONoCs components (Yan et al., 2020).

The two important factors that limit the scalability of ONoCs are crosstalk noise and power loss (Yan et al., 2020). Power loss and crosstalk noise are generated when an optical signal is transmitted from an input to an output of an optical element (Yan et al., 2020).

- Figure 2.9(a) shows a waveguide crossing. When the optical signal is transmitted through the waveguide crossing from the input port to port 2, the output power of port 2 (P_2), port 1 (P_1) and port 3 (P_3) can be calculated using formulas 2.5 and 2.6, respectively, where the crossing loss is represented by L_{cr}, the input power is represeneted by P_{in}, and the crosstalk coefficient of the crossing is represented by C_{cr} (Yan et al., 2020).

$$P_2 = L_{cr} P_{in} \tag{2.5}$$

$$P_1 = P_3 = C_{cr} P_{in} \tag{2.6}$$

- Figure 2.9(b) shows a waveguide bend. When an optical signal is transmitted through this optical element, the output power of port 1 (P_{out}) can be calculated using formula 2.7, where the bending loss and the input power are represented by L_b and P_{in}, respectively (Yan et al., 2020).

$$P_{out} = L_{be} P_{in} \tag{2.7}$$

- Figure 2.9(c) presents a PSE. When the optical signal is off-resonance with the MRR of the PSE, the optical signal is transmitted through the PSE from the input port to port Through, as can be seen in Figure 2.9(c). In this case, the output power of port Through ($P_{T,pse,off}$) can be calculated using formula 2.8, where the insertion loss is represented by $L_{p,off}$. In addition, the output power of port Drop $P_{D,pse,off}$ can be calculated using formula 2.9, where the crosstalk coefficient is represented by $C_{p,off}$ (Yan et al., 2020).

$$P_{T,pse,off} = L_{p,off} P_{in} \tag{2.8}$$

$$P_{D,pse,off} = C_{p,off} P_{in} \tag{2.9}$$

When the optical signal is on-resonance with the MRR of the PSE, the optical signal is transmitted through the PSE from the input port to port Drop, as can be seen in Figure 2.9(d). In this case, the output power of port Drop ($P_{D,pse,on}$) can be calculated using formula 2.11, where the insertion loss is represented by $L_{p,on}$. In addition, the output power of port Through ($P_{T,pse,on}$) can be calculated using formula 2.9, where the crosstalk coefficient is represented by $C_{p,on}$ (Yan et al., 2020).

$$P_{D,pse,on} = L_{p,on} P_{in} \tag{2.10}$$

$$P_{D,pse,on} = C_{p,on} P_{in} \tag{2.11}$$

- Figure 2.9(e) presents a CSE. As mentioned before, most designers only focus on using the 90-degree direction change. Therefore, this section presents only the power loss generated by the 90-degree rotation. When the optical signal is off-resonance with the MRR of the

PSE, the optical signal is transmitted through the CSE from the input port to port Through, as can be seen in Figure 2.9(e). In this case, the output power of port Through ($P_{T,cse,off}$) can be calculated using formula 2.12, where the insertion loss is represented by $L_{c_{,off}}$ (Yan et al., 2020). In addition, The output power of port Drop ($P_{D,cse,off}$) and port Add ($P_{A,cse,off}$) can be calculated by formulas 2.16 and 2.17, respectively.

$$P_{T,cse,off} = L_{c,off} P_{in} \tag{2.12}$$

$$P_{D,cse,off} = (C_{poff} + L_{poff}^2 C_{cr}) P_{in} \tag{2.13}$$

$$P_{A,cse,off} = L_{poff} C_{cr} P_{in} \tag{2.14}$$

When the optical signal is on-resonance with the MRR of the PSE, the optical signal is transmitted through the PSE from the input port to port Drop, as can be seen in Figure 2.9(f). In this case, the output power of port Drop ($P_{T,cse}$) can be calculated using formula 2.15, where the insertion loss is represented by $L_{c_{,on}}$ (Yan et al., 2020). In addition, the output power of port Drop ($P_{D,cseoff}$) and port Add ($P_{A,cseoff}$) can be calculated by formulas 2.16 and 2.17, respectively.

$$P_{D,cse,on} = L_{c,on} P_{in} \tag{2.15}$$

$$P_{T,cse,on} = C_{pon} L_{cr} P_{in} \tag{2.16}$$

$$P_{A,cse,on} = L_{pon} C_{cr} P_{in} \tag{2.17}$$

When the path consists of several optical elements, the power loss of the optical signal that transmits through the path can be calculated by the loss combination of all optical elements on the path (Yan et al., 2020). For example, when the optical signal is transmitted from I_0 to T_1 in the 4*3 GWOR Topology (see Figure 2.7), it goes through 3 CSE, specifically 1 ON MRR (a) and 2 OFF MRR. Therefore, the total insertion loss can be calculated by formula 2.18.

$$L_{21} = L_{c,on} L_{c,off}^2 \tag{2.18}$$

The signal-to-noise power ratio (SNR) can be used to evaluate the quality of the signals. The SNR for each signal x can be calculated using formula 2.19. The output power of a particular signal x, when this signal is transmitted from sender s to receiver r, is represented by $P^{\lambda_n}_{output}(x)$. The power of all crosstalk signals that reach the receiver r and have the same wavelength as the signal x is represented by $P^{\lambda_n}_{noise}(x)$ (Zheng et al., 2021). In this work, for simplicity, only the first-order noise introduced by the optical signals is considered.

$$SNR(x) = 10log\frac{P^{\lambda_n}_{output}(x)}{P^{\lambda_n}_{noise}(x)} \tag{2.19}$$

—

3. Related Work

Several research about maximizing the communication parallelism for WRONoCs were published in the past decade. The first section presents the approach published by (Li et al., 2020) which maximizes the parallelism at the stage of MRR-radius selection. The second section presents the approach published by (Chen et al., 2023), which aims to maximize the worst case parallelism at the phase of topology generation.

3.1. Maximization of the Communication Parallelism

Most ONoCs studies ignore communication parallelism (Li et al., 2020). In these studies, each master can communicate with each of its slaves with only one specific wavelength (Li et al., 2020). Therefore, Li *et al.* (2020) proposed an integer-linear-programming (ILP) formulation to increase the communication parallelism of each path in a given topology by assigning several wavelengths to each path (Li et al., 2020). This work performs better than the previous approaches (such as (Peano et al., 2016)) in terms of improving the parallelism (Li et al., 2020). The method in (Li et al., 2020) is used to assign a radius to each MRR type in the proposed topology. This section presents the implementation of this ILP model using gurobipy in a comprehensive manner.

3.1.1. Scoping and Problem Formulation

3.1.1.1. Input

- The modeling approach presented by (Li et al., 2020) works only for a given topology. Therefore, the signal paths of a topology must be specified. Each path should indicate which on- and off-resonance MRR types it contains. For example, if the 4*3 GWOR topology is used (see Figure 2.7), the path from I_0 to T_1: should be specified as follows:

  $I_0 \rightarrow T_1$: $ON_{[I_0,T_1]} = \{MRR_a\}$; $OFF_{[I_0,T_1]} : \{MRR_b\}$.

- The MRR-radii options should be provided. The resonant wavelengths that match an MRR radius can be calculated using the approaches presented in (Peano et al., 2016) and (Parini et al., 2011).

3.1.1.2. Output

- One radius should be assigned to each MRR type in the topology (Li et al., 2020).

- The values of the wavelengths should be assigned to each path in the topology (Li et al., 2020).

3.1.2. The Approach

As mentioned before, this method works only for a given topology. A Python list M is used to present the various types of MRRs used in the topology, for example, M=[a, b]. A Python dictionary (named Path) is used to denote the signal paths. Two keys are used to define each path. The first key denotes the on-resonance MRRs in the path. The second key denotes the off-resonance MRRs in the path. For example, in order to get the off-resonance MRRs in the path from I_0 to T_1, the key should be ['I0','T1','OFF'], where the first and the second items of the key represent the master and the slave, respectively.

To define the paths of the 4*3 GWOR topology (see Figure 2.7), the following Python Dictionary should be used, where the elements of the M list are used as the dict values:

```
PATH = {
    ('I0', 'T1', 'ON'): ['a'],
    ('I0', 'T1', 'OFF'): ['b'],
    ('I0', 'T2', 'ON'): ['b'],
    ('I0', 'T2', 'OFF'): [],
    ('I0', 'T3', 'ON'): [],
    ('I0', 'T3', 'OFF'): [],
    ('I1', 'T0', 'ON'): ['a'],
    ('I1', 'T0', 'OFF'): [],
    ('I1', 'T2', 'ON'): [],
    ('I1', 'T2', 'OFF'): [],
    ('I1', 'T3', 'ON'): ['b'],
    ('I1', 'T3', 'OFF'): ['a'],
    ('I2', 'T0', 'ON'): ['b'],
```

```
('I2', 'T0', 'OFF'): ['a'],
('I2', 'T1', 'ON'): [],
('I2', 'T1', 'OFF'): [],
('I2', 'T3', 'ON'): ['a'],
('I2', 'T3', 'OFF'): [],
('I3', 'T0', 'ON'): [],
('I3', 'T0', 'OFF'): [],
('I3', 'T1', 'ON'): ['b'],
('I3', 'T1', 'OFF'): [],
('I3', 'T2', 'ON'): ['a'],
('I3', 'T2', 'OFF'): ['b']}
```

In addition, two lists are introduced which represent the sets of masters and slaves. These two lists are used to access and iterate the Path Dict:

```
I = ['I0', 'I1', 'I2' , 'I3']
T = ['T0', 'T1', 'T2' , 'T3']
```

Furthermore, a set of MRR-radii options is denoted by R, for example, $R = \{r_1, r_2, r_3, r_4, r_5, ...\}$ or $R = \{5, 5.25, 5.5, 5.75, 6.0,...\}$ (Li et al., 2020). In this Python implementation, a list R is used to represent the MRR-radii options, for example:

```
R= [5, 5.25, 5.5, 5.75, 6.0]
```

Additionally, a sequence Ω_r is constructed to represent the corresponding resonant wavelengths of r. $\Omega_r^{(i)}$ represents the i-th resonant wavelength in the sequence Ω_r. For example, if $R = \{ r_1, r_2\}$ is the set of MRR-radii options, the sequence of resonant wavelengths of each MRR-radii option can be represented as follows:
$\Omega_{r_1} = \{\lambda_1, \lambda_3, \lambda_5, \lambda_6\}$, which means: $\Omega_{r_1}^{(1)} = \lambda_1$, $\Omega_{r_1}^{(2)} = \lambda_3$, $\Omega_{r_1}^{(3)} = \lambda_5$, $\Omega_{r_1}^{(4)} = \lambda_6$.
$\Omega_{r_2} = \{\lambda_2, \lambda_4, \lambda_6\}$, which means: $\Omega_{r_2}^{(1)} = \lambda_2$, $\Omega_{r_2}^{(2)} = \lambda_4$, $\Omega_{r_2}^{(3)} = \lambda_6$.
In the Python implementation, a Python dictionary (named Omiga) is used to represent the resonant wavelengths for each MRR-radius, where each MRR-radius represents a key value. The corresponding resonant wavelengths of r are the values of the dict and are stored in a Python list. Here is an example of the Omiga Dict (Tala & Bertozzi, 2018):

```
Omiga = {
5    : [1496.4 , 1521.3 , 1547.1 , 1573.8 , 1601.4],
5.25: [1497.5 , 1521.3 , 1545.9 , 1571.2 , 1597.4]
}
```

In this example, Omiga [5.25] represents $\Omega_{5.25}$ and gives the resonant wavelengths for r =

5.25 stored in a list: [1497.5 , 1521.3 , 1545.9 , 1571.2 , 1597.4]. Omiga[5.25][0] represents $\Omega_{5.25}^{(0)}$=1497.5.

3.1.3. Creating the Gurobi Model

First, the Gurobi functions and classes need to be imported (GUROBI, 2020).

```
import gurobipy as gp
from gurobipy import GRB
```

Then, the objective model is created. A Gurobi model consists of a set of constraints, a set of integer, binary and continuous variables and several associated attributes, such as the objective coefficients and the variable bounds. In addition, it contains a single optimization problem. The function Model() is used to create the model. The desired model name is written as an argument of this function:

```
m = gp.Model('MA')
```

3.1.4. The Assignment of Radii

First, a binary variable b_r^m is introduced, for each m and r in M and R, respectively (Li et al., 2020). If 1 is assigned to this variable, it indicates that the radius r is assigned to the MRRs of type m (Li et al., 2020). In the Python implementation, the function *addVars* of the Gorubi library is used to introduce variables. To generate a binary variable, the *vtype* should be set to *GRB.BINARY*. $|R| \times |M|$ variables should be declared. Therefore, The first and the second arguments are set to R and M, respectively. Thus for each pair (r, m), where r ∈ R and m ∈ M, a variable brm[r,m] is created. Therefore, the lists M and R are used to iterate the variables brm. The declaration of these variables can be done by the following command:

```
brm = m.addVars(R, M, vtype=GRB.BINARY, name="brm")
```

Afterwards, the constraint 1 is introduced in order to assign exactly one radius to each MRR type (Li et al., 2020).

$$\sum_{r\in R} b_r^m = 1 \qquad \forall m \in M \tag{1}$$

Each two MRRs of different types can not have the same radius. Therefore, constraint 2 is introduced in order to ensure that each radius $r \in R$ can be assigned to at most one MRR-type.

$$\sum_{m \in M} b_r^m \leq 1 \qquad \forall r \in R \tag{2}$$

These constraints can be implemented with Gurobi using the function addConstrs. This function takes the constraint and the name of the constraint as its first and second arguments, respectively. For all $m \in M$, $\sum_{r \in R} b_r^m = 1$ should be fulfilled. Therefore, in total there are $|M|$ constraints. In Gurobi, it is possible to use a for loop within the first argument of the addConstrs (for m in M) in order to implement the $\forall$ symbol ($\forall m \in M$). Therefore, these constraints can be implemented using the following commands:

```
m.addConstrs((brm.sum('*',mm) == 1 for mm in M), "c1")

m.addConstrs((brm.sum(r,'*') <= 1 for r in R), "c2")
```

3.1.5. The Assignment of Wavelength

A path is called a default path, when it does not contain any on-resonance MRR. In this case, all of the MRRs in this path are OFF. Therefore, Multiple wavelengths can be found that do not conflict with the OFF-MRRs in this path. On the one hand, if the path p contains one on-resonance MRR, only the resonant wavelengths supported by this on-resonance MRR are considered at the beginning of the wavelength selection process for path p. Then, the wavelengths that conflict with the OFF-MRRs are removed. On the other hand, in the case of the default path, there is no constraint on the wavelengths that can be considered at the beginning. Finding several wavelengths for the default path can be done after assigning a radius to each MRR type in the topology. Thus, the ILP model aims to assign a radius to each MRR type in the topology and to find the wavelengths for all paths other than the default paths. The names of the paths that have at least one ON MRR are stored in the list P.

The constant n is introduced to store the number of the resonant wavelengths supported by the MRR-radius that have the highest number of the resonant wavelengths among all MRR-radius options, i.e., $n = max_{r \in R}|\Omega_r|$ (Li et al., 2020). Then, n continuous variables w_k^p are introduced for each path that contains at least one on-resonance MRRs (Li et al., 2020). These variables of each path p are used to store the exact wavelength value assigned to the path p. The lower bound of each of these variables is 0, which indicates an empty placeholder. The given wavelength band defines the upper bound of the variables w_k^p. When a wavelength is assigned to path p, the exact value of a wavelength is stored in one of these variables of the path p (w_k^p).

In the Python implementation, to generate a continuous variable, the *vtype* should be set to *GRB.CONTINUOUS*. To set an upper bound, the *ub* should be set to the desired upper bound value. Similarly, to set a lower bound, the *lb* should be set to the desired lower bound value. For each pair (x,y), where x $\in$ P and y $\in \{0, 1, 2, ..., n-1\}$, a variable w[x,y] is created. Therefore, The first and the second arguments are set to P and n, respectively. The declaration of these variables (n variable for each path that has one ON MRR) can be done by the following command:

```
w = m.addVars(P,n,vtype=GRB.CONTINUOUS,ub=1600, lb=0, name="w")
```

The following constraints are introduced to assign all the wavelengths of a radius r (Ω_r) to the wavelengths w_k^p of the path p if the radius r is selected by an on-resonance MRR in this path:

$$\forall p \in P \quad \forall r \in R \quad \forall 1 \leq k \leq |\Omega_r|$$

$$\mathrm{w}_k^p + \textstyle\sum_{m \in ON_p} b_r^m.M \leq \Omega_r^{(k)} + M$$

$$\mathrm{w}_k^p - \textstyle\sum_{m \in ON_p} b_r^m.M \geq \Omega_r^{(k)} - M \qquad \text{(C)}$$

Afterwards, two auxiliary binary variables for each radius r and for each wavelength w of each path are generated in order to bring the off-resonance MRR into consideration (Li et al., 2020). Therefore, the constraints in C are modified (Li et al., 2020):

$$\forall p \in P \quad \forall r \in R \quad \forall 1 \leq k \leq |\Omega_r|$$

$$w_k^p + \sum_{m \in ON_p} b_r^m.M \leq \Omega_r^{(k)} + q_{1,p,r,k}.M + M \tag{3}$$

$$w_k^p - \sum_{m \in ON_p} b_r^m.M \geq \Omega_r^{(k)} - q_{1,p,r,k}.M - M \tag{4}$$

$$w_k^p + \sum_{m \in ON_p} b_r^m.M \leq q_{2,p,r,k}.M + M \tag{5}$$

$$w_k^p - \sum_{m \in ON_p} b_r^m.M \geq -q_{2,p,r,k}.M - M \tag{6}$$

$$q_{1,p,r,k} + q_{2,p,r,k} = 1 \tag{7}$$

After this modification, if $\sum_{m \in ON_p} b_r^m = 0$, these constraints are tautologies. Otherwise, if $\sum_{m \in ON_p} b_r^m = 1$, w_k^p depends on the value of $q_{1,p,r,k}$ and $q_{2,p,r,k}$ (Li et al., 2020):

$$w_k^p = \begin{cases} \Omega_r^{(k)}, & q_{1,p,r,k} = 0 \wedge q_{2,p,r,k} = 1 \\ 0, & q_{1,p,r,k} = 1 \wedge q_{2,p,r,k} = 0 \end{cases}$$

Constraint 7 ensures that either $q_{1,p,r,k}$ or $q_{2,p,r,k}$ is 1. If the wavelength $\Omega_r^{(k)}$ is in conflict with an off-resonance MRR in the path p, $q_{2,p,r,k}$ will be set to 0, which prevents $w_p^{(k)}$ from receiving the value $\Omega_r^{(k)}$ according to constraints 3, 4, 5, 6 and 7 (Li et al., 2020). If $\Omega_r^{(k)}$ is not in conflict with any resonant wavelengths supported by all off-resonance in the path, 1 is assigned to $q_{2,p,r,k}$ (Li et al., 2020).

$\forall p \in P \quad \forall r \in R \quad \forall 1 \leq k \leq n$, 2 auxiliary binary variables are introduced. Therefore, 2*(pa*om*n) additional binary variables are introduced, where pa is the number of paths with one ON MRR and om is the number of MRR-radii options. In the Python implementation, these additional binary variables can be implemented using the following command:

```
q1 = m.addVars(pa*om*n, vtype=GRB.BINARY, name="q1")

q2 = m.addVars(pa*om*n, vtype=GRB.BINARY, name="q2")
```

As mentioned in Section 2.3, two wavelengths conflict with each other when the absolute difference between them is smaller than the safe distance δ (Li et al., 2020). A radius r conflicts with wavelength Ω, if at least one resonant wavelength supported by the radius r conflicts with Ω (Li et al., 2020). Therefore, all the MRR-radii options conflicting with $\Omega_r^{(k)}$ are collected into a set $R_{\Omega_r^{(k)}}^{conflict}$ (Li et al., 2020). The constraint 8 is set in order to link $q_{2,p,r,k}$ to the off-resonance MRR in the path p (Li et al., 2020). If the wavelength $\Omega_r^{(k)}$ conflicts with an off-resonance MRR in the path p, $\sum_{r' \in R_{\Omega_r^{(k)}}^{conflict}} b_{r'}^{m'} = 1$, $q_{2,p,r,k}$ is set to 0, which prevents $w_p^{(k)}$ from receiving the value $\Omega_r^{(k)}$ according to constraints 3, 4, 5, 6 and 7 (Li et al., 2020).

$$\forall m' \in OFF_p \qquad \sum_{r' \in R_{\Omega_r^{(k)}}^{conflict}} b_{r'}^{m'} + q_{2,p,r,k} \leq 1 \tag{8}$$

To implement this constraint, a function *conflict (x, r)* is created. This function takes the wavelength and the corresponding radius and returns the radii that conflict with this

wavelength. For example, here are some radii options (Tala & Bertozzi, 2018), if $R = \{7.25, 7.5, 7.75\}$ and
$\Omega_{7.25}$: [1504.0, 1521.3, 1539.0, 1557.1, 1575.6, 1594.6],
$\Omega_{7.5}$: [1504.6, 1521.3, 1538.4, 1555.9, 1573.8, 1592.1, 1610.8],
$\Omega_{7.75}$: [1505.1, 1521.3, 1537.9, 1554.8, 1572.0, 1589.7, 1607.8].
The safe distance is $\delta = 0.8$ nm. In this case, the function *conflict (1504.6, 7.5)* returns *[7.25, 7.75]*, which means $R^{conflict}_{\Omega^{(0)}_{r=7.5}} = R^{conflict}_{\Omega_{1504.5}} = \{7.25, 7.75\}$ because the first wavelegth of $r = 7.25$ ($\Omega^{(0)}_{7.25} = 1504.0$) and r = 7.75 ($\Omega^{(0)}_{7.75} = 1505.1$) conflicts with the wavelength 1504.5 of the radius $r = 7.5$. On the other hand, the function *conflict (1575.6, 7.25)* returns [], which means $R^{conflict}_{\Omega^{(4)}_{r=7.25}} = R^{conflict}_{\Omega_{1575.6}} = \emptyset$ because the absolute difference between the wavelength $\Omega^{(4)}_{r=7.25} = 1575.6$ and each wavelength of the radii $r = 7.5$ and $r = 7.75$ is always larger than $\delta = 0.8$ nm. The function *conflict (x, r)* is used to find and to store the radii that are in conflict with each wavelength of each radius in a list Conflict. Like the other constraints, constraint 8 can be implemented using the function addConstrs:

```
m.addConstrs((
gp.quicksum(brm[rp, mm] for rp in Conflict[R.index(r)][omiga[r].index(x)])
+ q2[P.index(ii) * om * 23 + R.index(r) * 23 + omiga[r].index(x)] <= 1
for ii in P # for all paths
for mm in PATH[ii[0:2], ii[2:], 'OFF']
for r in R # for all r
for x in omiga[r])
, "c8")
```

In addition, as mentioned before, for each path p there is n continuous variables representing wavelength placeholders. The constraints 9 and 10 are added to assign 0 to the redundant placeholders (Li et al., 2020).

$$\forall p \in P \quad \forall r \in R \quad \forall |\Omega_r| + 1 \leq k \leq n :$$

$$w^p_k + \sum\nolimits_{m \in ON_p} b^m_r.M \leq M \tag{9}$$

$$w^p_k - \sum\nolimits_{m \in ON_p} b^m_r.M \geq -M \tag{10}$$

For each path and each radius r, if r is assigned to any on-resonance MRR in the path p, $\mid \Omega_r \mid$ values can be assigned to the first $\mid \Omega_r \mid$ w^p_k ($1 \leq k \leq\mid \Omega_r \mid$) values according to constraints 3, 4, 5, 6 and 7, but the last $n-\mid \Omega_r \mid$ values are forced to take 0 according to constraints 9 and 10. For example, $n = 5$ and $\Omega_{r_1} = \{\lambda_2, \lambda_3, \lambda_5\}$, which means: $\Omega^{(0)}_{r_1} = \lambda_2$, $\Omega^{(1)}_{r_1} = \lambda_3$, $\Omega^{(2)}_{r_1} = \lambda_5$. The path from I_1 to T_0 contains only one on-resonance MRR ('a') and no off-resonance MRR. If '1' is assigned to $b^a_{r_1}$, then $\Omega^{(0)}_{r_1}$, $\Omega^{(1)}_{r_1}$ and $\Omega^{(2)}_{r_1}$ will be assigned to $w_0^{I_1-T_0}$, $w_1^{I_1-T_0}$ and $w_2^{I_1-T_0}$, respectively, according to constraints 3, 4, 5, 6 and 7. In addition, according to constraints 9 and 10, 0 will be assigned to $w_3^{I_1-T_0}$ and $w_4^{I_1-T_0}$.

3.1.6. The Objective

The target is to maximize the aggregate parallelism (Li et al., 2020). When w^p_k takes a non zero value, it means that a wavelength value is assigned to this path. Thus, the number of w^p_k that take a non zero value needs to be maximized in order to achieve the target. Therefore, a binary variable b^P_k is introduced for each wavelength w^p_k (Li et al., 2020). This binary variable takes a value of 1 when the wavelength w^p_k takes a non-zero value (Li et al., 2020). Thus, constraint 11a is introduced (Li et al., 2020). As can be noted, if w^p_k is 0, b^P_k takes 0. Otherwise, if $w^p_k \geq 1$, b^P_k can take either 0 or 1. Considering the optimization objective which aims to maximize the sum of the values of b^P_k, b^P_k will be forced to take 1 in the case of having $w^p_k \geq 1$. The integer variable v_{total} is introduced to represent the sum of the wavelengths assigned to each path, as can be seen in Equation 11b (Li et al., 2020):

$$\forall p \in P \quad \forall 1 \leq k \leq n : \qquad b^p_k \leq w^p_k \tag{11a}$$

$$vtotal = \sum_{p \in P} \sum_{1 \leq k \leq n} b^p_k \tag{11b}$$

This constraint can be implemented in Gurobi using the following commands:

```
bpk = m.addVars(P,n, vtype=GRB.BINARY, name="bpk")
m.addConstrs( (bpk[i,k] <= w[i,k] for i in P
for k in range(n)), "c11")
Vtotal = m.addVar(vtype=GRB.INTEGER,name="Vtotal")
m.addConstr( ( Vtotal <= bpk.sum('*', '*') ), "c13")
```

Another target is maximizing the worst case parallelism (Li et al., 2020). Therefore, the integer variable v_{worst} is introduced (Li et al., 2020). Constraint 12 is added to ensure that v_{worst} is smaller or equal to the number of wavelengths assigned to each path (Li et al., 2020). Considering the optimization objective which can be set to maximize the worst case signal path parallelism, v_{worst} will be forced to take the lowest value of $\sum_{1 \leq k \leq n} b^p_k, \forall p \in P$ (Li et al., 2020).

$$\forall p \in P : \qquad v_{worst} \leq \sum\nolimits_{1 \leq k \leq n} b_k^p \qquad (12)$$

Similar to the previous constraint, the constraint 12 can be implemented with the following commands:

```
Vworst = m.addVar(vtype=GRB.INTEGER, name="Vworst")
m.addConstrs( ( Vworst <= bpk.sum(iii, '*')for iii in P ), "c12")
```

The coefficients α and β are used to formulate the optimization model. The optimization model can focus on improving the worst signal path parallelism by setting the variable α to 1 and the variable β to 0, or maximizing the sum of wavelengths assigned to all paths by setting α to 0 and β to 1, or can focus on both optimization targets by setting, for example, $\alpha = 10$ and $\beta = 1$ (Li et al., 2020).

$$Maximize : \alpha . v_{worst} + \beta . v_{total}$$

Subject to: (1) - (12)

The objective is built in Gurobi using the function setObjective. The first argument indicates the objective, whereas the second argument represents the sense of the optimization: whether it is maximization or minimization. Then, the last step is to optimize the model using the function m.optimize ().

```
alpha = 10
beta= 1
m.setObjective( alpha*Vworst + beta*Vtotal , GRB.MAXIMIZE)
m.optimize ()
```

3.1.7. Example

In this section, the approach proposed by (Li et al., 2020) is used to maximize the parallelism in the 4x3 GWOR topology (see Figure 2.7). The paths in this topology are presented in Subsection 3.1.2. There are two MRR-types: *'a' and 'b'*. The upper bound of the wavelength is 1600 nm. Several MRR-radii options between 5 μm and 7.25 μm are provided (Tala & Bertozzi, 2018):
5 : [1496.4 , 1521.3 , 1547.1 , 1573.8 , 1601.4],
5.25: [1497.5 , 1521.3 , 1545.9 , 1571.2 , 1597.4],
5.5 : [1498.6 , 1521.3 , 1544.7 , 1568.9 , 1593.8],

5.75: [1499.6 , 1521.3 , 1543.7 , 1566.7 , 1590.5],
6 : [1500.5 , 1521.3 , 1542.7 , 1564.8 , 1587.5 , 1610.8],
6.25: [1501.3 , 1521.3 , 1541.9 , 1563.0 , 1584.7 , 1607.0],
6.5 : [1502.1, 1521.3, 1541.1, 1561.4, 1582.2, 1603.5],
6.75 :[1502.8, 1521.3, 1540.3, 1559.8, 1579.8, 1600.3],
7.0 :[1503.4,1521.3,1539.6,1558.4,1577.7,1597.4],
7.25: [1504.0,1521.3, 1539.0,1557.1,1575.6,1594.6]
α and β were both set to 1.

In order to obtain the results, The function getVars() is used. In addition, the variable attributes varName and x are used to get the name and solution value for each variable, respectively (GUROBI, 2020):

After running the python implementation of this module, the radii 6.5 and 7 are assigned to MRRs of type 'b' and 'a', respectively. V_{total} was 40 and V_{worst} was 4 in the paths from *I1* to *T3* and from *I2* to *T0*. The wavelength values selected for each path are stored in the w variables of this path. For example:
w[I1T3,0] = 1502.1
w[I1T3,1] = 0
w[I1T3,2] = 1541.1
w[I1T3,3] = 1561.4
w[I1T3,4] = 1582.2
w[I1T3,5] = 0
w[I1T3,6] = 0

As can be seen, in this case, the wavelength 1521.3 is not selected because it conflicts with the radius r=7. In addition, the wavelength 1603.5 is not selected because it is larger than the upper bound of w_k^p: 1600 .

3.1.8. Discussion

Using more wavelengths leads to more crosstalk noise as well as more laser power consumption. To reduce the impact of these concerns, the spacing between wavelengths can be increased (Li et al., 2020). Considering that in practice MRR radius options are limited, the number of wavelengths assigned to each path would be very small or infeasible in some WRONoC topologies. For example, assume there are only two radius options r_1 and r_2: $\Omega_{r_1} = \{\lambda_{11} =$

1531, $\lambda_{12} = 1586\}$ and $\Omega_{r_2} = \{\lambda_{21} = 1531.5, \lambda_{22} = 1552, \lambda_{23} = 1585.5, \lambda_{24} = 1599.5\}$. The safe distance is 0.8. Figure 3.1(a) represents the used topology (The CSEs support 270-degree as well as 90-degree direction changes of optical signals.). There are five signals (c1, c3), (c1, c5), (c1, c4), (c2, c5) and (c6, c5). As can be seen, the signal (c1, c5) should be on-resonance with the red MRR and off-resonance with the blue MRR. In addition, the signal (c1, c4) should be on-resonance with the blue MRR and off- resonance with the red MRR. Assume r_1 is assigned to the red MRR and r_2 is assigned to the blue MRR. Since $|\lambda_{11} - \lambda_{21}| < 0.8$ and $|\lambda_{11} - \lambda_{21}| < 0.8$, there is no wavelength that is on-resonance with r_1 and off-resonance with r_2. Thus, core c1 could not communicate with core c5. Therefore, the worst case parallelism in this case is 0. Similarly, if r_1 is assigned to the blue MRR and r_2 is assigned to the red MRR, core c1 could not communicate with core c4. Therefore, for this topology, there is no practical MRR type mapping. However, if the parallelism is considered during topology generation, the topology in Figure 3.1(b) could be a possible output. In this case, signal (c1, c5) should be on-resonance with the red MRR and off- resonance with the blue MRR. In contrast to the topology in 3.1(a), signal (c1, c4) should only be on-resonance with the blue MRR. In this case, if r_1 is assigned to the blue MRR and r_2 is assigned to the red MRR, core c1 could communicate with core c5 with 2 wavelengths $(\lambda_{22}, \lambda_{24})$. In this case, the worst case parallelism is 2 and the aggregate parallelism is 6. In conclusion, considering the communication at the stage of topology generation leads to increasing the parallelism.

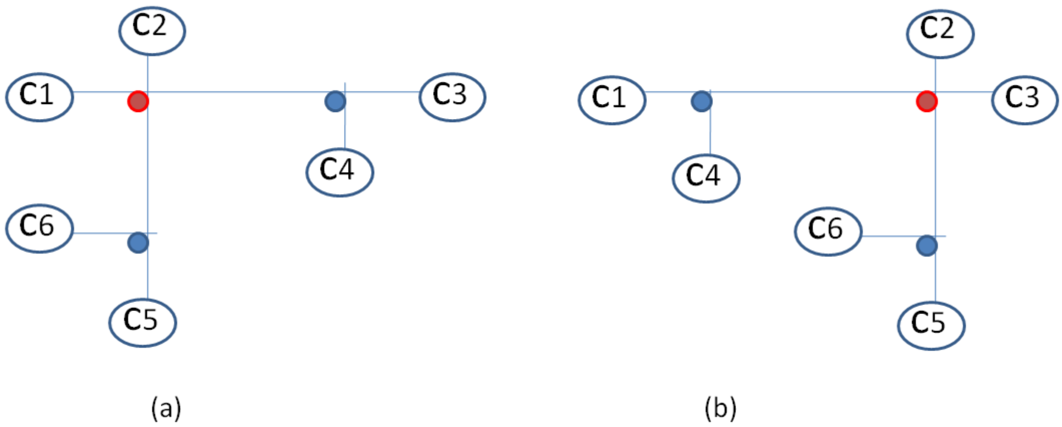

Figure 3.1.: Two simple topologies

It is possible to note that the more off-resonance MRR are in a path, the fewer wavelengths are assigned to this path. Thus, this master's thesis presents an approach for designing a topology that takes into account the communication at the design phase by minimizing the sum of the numbers of OFF MRR types in all paths.

3.2. Optimizing the Worst Case Parallelism

Chen *et al.* (2023) presented a method that aims to improve the worst case parallelism at the phase of topology generation (Chen et al., 2023). Chen *et al.* (2023) indicated that their work is optimal in terms of maximizing the worst case parallelism for fully connected N × (N-1) topology (Chen et al., 2023). Chen *et al.* (2023) demonstrated that minimizing the MRR type usage leads to optimal worst-case parallelism, if the topology satisfies Property 3.1. In Property 3.1, m_i (i $\in \mathbb{Z}^+$) represents the MRR type i. The MRR types are M={m_1, m_2, $m_3, ...$}. The set P represents the set of all paths. The set ON_p represents the type of the MRR that path p activates. The set OFF_p represents the types of the MRRs that path p must not activate. In fact, Property 3.1 is equivalent to Property 3.2.

$$\forall p \in P \quad \forall i, j \in \mathbb{Z}^+ : ((i < j) \;\; \& \;\; (m_i \in ON_p)) \Longrightarrow m_j \notin OFF_p \tag{3.1}$$

$$\forall k \in \mathbb{Z}^+ : m_k \in ON_p \Longrightarrow OFF_p \subseteq \{m_1, m_2, ..., m_{k-1}\} \tag{3.2}$$

For example, assume there are 3 MRR types in the topology: m_1, m_2 and m_3. To satisfy Property 3.1, constraints 3.3, 3.4 and 3.5 must be satisfied for each path p.

$$m_1 \in ON_p \Longrightarrow OFF_p \subseteq \emptyset \tag{3.3}$$

$$m_2 \in ON_p \Longrightarrow OFF_p \subseteq \{m_1\} \tag{3.4}$$

$$m_3 \in ON_p \Longrightarrow OFF_p \subseteq \{m_1, m_2\} \tag{3.5}$$

This method has two steps, first constructing a network, and second minimizing the MRR type usage using integer linear programming (ILP) (Chen et al., 2023). For the first step “Network Construction”, Figure 3.2 represents a general 4×4 template for the topology that satisfies Property 3.2 (Chen et al., 2023). The template has n cores along the diagonal. Each core has two ports: Input and Output. For all n × n-1 communication matrices, with this template, it is always possible to find a topology that satisfies Property 3.1 using at most n-1 MRR types (Chen et al., 2023).

The second step aims to select the core positions on the template and to assign a type to each MRR, while satisfying Property 3.1, using ILP. The target of the ILP is to reduce the

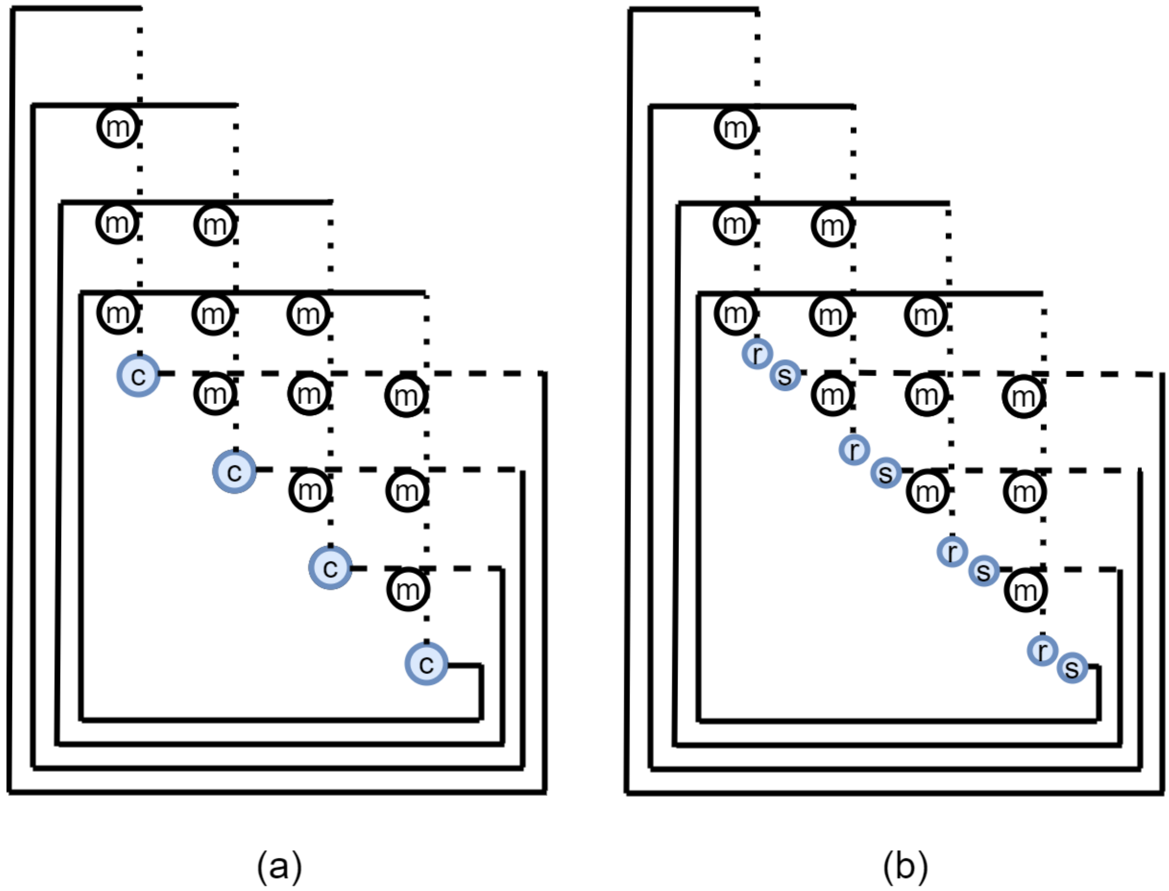

Figure 3.2.: The template for 4x4 Network: (a) with cores (b) with senders and receivers (Chen et al., 2023).

number MRR types used (Chen et al., 2023). The selection of the default paths in the topology depends on the position of the cores. Here are the main ideas of the working mechanism of the ILP mode:

For each connection (s, r), an integer variable $m_{s,r}$ is introduced to represent the type of the MRR that path (s, r) activates (Chen et al., 2023). For example, $m_{s,r}$ = i means that path (s, r) is on-resonance with MRR type m_i. An integer variable K is introduced to represents the upper bound of the variables $m_{s,r}$ (Chen et al., 2023):

$$1 \leq m_{s,r} \leq K \qquad \forall (s, r) \in P \tag{3.6}$$

The target of the ILP mode is to minimize the value of K (Chen et al., 2023). The main idea of the method is to iterate through all possible core placements. For each iteration, according to the position of the cores, the maximization target, and to Property 3.1, a value is assigned to each $m_{s,r}$. According to the position of the MRRs on the template, for each MRR (s, r) (activated by path (s, r)), the value K is assigned to $m_{s,r}$, if there are no MRRs above or to the left of MRR (s, r) on the template.

Finally, when the ILP model terminates, the outputs of ILP mode are used to select the core positions on the template. The value of $m_{s,r}$ are used to select the MRR positions and their types. Finally, each MRR (s, r) of type m_K ($m_{s,r}$ = K) is transformed into a default path (The horizontal waveguide and the vertical waveguide attached to MRR (s, r) are merged as one and the MRR (s, r) is removed). Figure 3.3 presents an example of designing the topology. First, depending on the output of the ILP model, the position of the MRRs and the cores are arranged, as can be seen in Figure 3.3 (a). Then, the MRRs of types $m_{s,r}$ = K (K=3) are transformed into a default paths, as can be seen in Figure 3.3 (b).

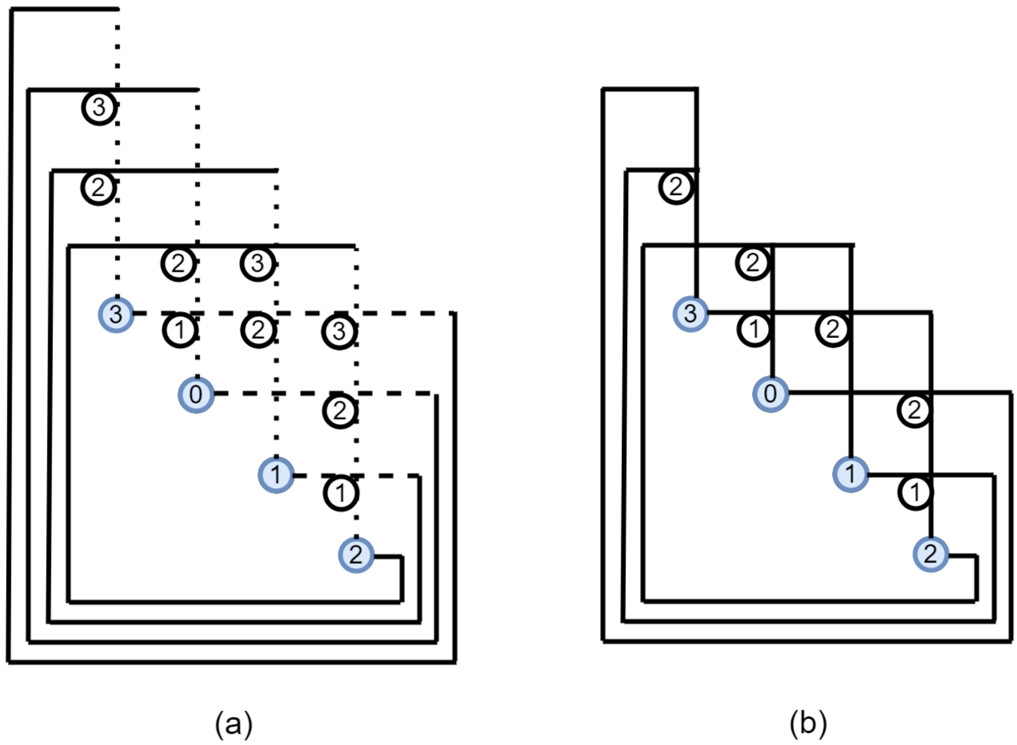

Figure 3.3.: An example of designing the topology in the previous work.

Experimental results shows that for a fully connected graph this topology performs better than the GWOR topology in terms of maximizing the worst case parallelism by 51.5% and minimizing the insertion loss by 4.3% (Chen et al., 2023).

3.2.1. Discussion

As mentioned in Subsection 3.1.8, the more off-resonance MRR types are in a path, the fewer wavelengths can be assigned to this path. The work in (Chen et al., 2023) aims only to maximize the worst case parallelism by reducing the MRR type usage and satisfying Property 3.2. Therefore, it does not focus on minimizing the number of the OFF MRR types in the paths other than the paths with the highest number of OFF MRR types. For example, it does not aim to maximize the number of default paths and the number of paths with one ON MRR and zero OFF MRRs.

Since the template in Figure 3.2 has n cores along the diagonal, the number of paths with one ON MRR and 0 OFF MRRs can be n. Similarly, the number of paths with one ON MRR and one OFF MRR can be n. Therefore, putting the cores along the diagonal is a good starting idea to maximize the aggregate parallelism.

The work in (Chen et al., 2023) does not focus on maximizing the number of default paths. It aims only to transform the paths with the largest number of OFF MRRs to default paths. In the work in (Chen et al., 2023), the selection of the default paths depends on the position of the cores. As mentioned in the previous subsection, the MRRs that have type K are transformed into a default path. Therefore, the number of default paths is in many cases not maximized. This inflexible selection of default paths could limit the improvement in worst case parallelism.

Additionally, the topology does not support self-connectivity, i.e., the sender i is not connected to the receiver i. It works only for n $\times$ (n-1) communication matrices.

4. The Proposed Topology

The wavelength conflict for each path should be reduced in order to improve the parallelism of a topology. Therefore, the number of different OFF MRR types in each path should be minimized. In addition, the number of default paths should be maximized.

4.1. Scoping and Problem Formulation

4.1.1. Input

- The communication matrix C: The input matrix C (Binary matrix) represents the connections between the senders and the receivers. Each row index represents a sender (master). Each column index represents a receiver (slave). There is a connection between sender s and receiver r (sender s sends data to receiver r), if the entry (s, r) in the communication matrix is equal to 1. The size of the communication matrix is $\eta \times \eta$.

4.1.2. Output

- An image of the topology:
  - Each MRR is colored with a color according to its type. Each type has its specific color (The MRRs of the same type are colored with the same color).
  - The waveguides are represented in the image as vertical and horizontal lines.
  - Each core is represented in the image as a number in a circle. The core i is represented in the image by the number i in a circle.
- A file contains information about the topology:

- The MRR types used in the topology.
- The on- and off-resonance MRR types of each path.

4.1.3. Maximization Objective

- Maximizing the worst case parallelism.
- Improving the aggregate parallelism.
- Reducing the insertion loss.

4.2. The Topology

4.2.1. The Improvement of Parallelism

The set of MRRs are represented by $M = \{m_1, m_2, m_3, ...\}$. The set of paths is represented by P. For each path $p \in P$, the sets ON_p and OFF_p represent the on- and off-resonance MRRs of path p, respectively. For each path, the set ON_p can either be an empty set or contains one element $|ON_p| \leq 1$ (Each path can activate at most one MRR).

As mentioned in Subsection 3.1.8, the fewer off-resonance MRR types are in a path, the more wavelengths can be assigned to this path. Therefore, this master's thesis presents an approach for generating a topology that improves the aggregate parallelism in the design phase by reducing the MRR types used in each path. This approach also aims to maximize the worst case parallelism in the design phase by minimizing the MRR types used in the path with the highest number of MRR types.

To improve the aggregate parallelism, the proposed method attempts to satisfy Property 4.1, while minimizing the sum in Equation 4.2 and maximizing the number of default paths (D). Equation 4.2 aims to minimize a weighted combination of several individual objectives (N_i represents the number of used m_i). Equation 4.3 represents the relationship between the number of default paths and the number of used MRRs of each type. In fact, the proposed method aims to minimize Equation 4.2 and maximize the number of default paths while satisfying Equation 4.3 and Property 3.1 (Chen et al., 2023).

$$\forall p \in P \quad \forall x \in \mathbb{N} : m_x \in ON_p \Longrightarrow OFF_p = \{m_{x-1}, m_{x-2}, ...m_1\} \tag{4.1}$$

$$\text{minimize} \sum_{i \in \{1,2,..,\eta\}} i * N_i \tag{4.2}$$

$$\sum_{i \in \{1,2,..,\eta\}} N_i + D = \sum_{i \in \{1,2,..,\eta\}} \sum_{j \in \{1,2,..,\eta\}} C[i][j] \tag{4.3}$$

4.2.2. Network Template

The topology design process includes four main steps, as can be seen in Figure 4.1. The first step is to construct a ***network template***. This step consists of two main phases: The first phase presents the network template without default paths. In this phase, the positions of the cores and the potential MRR positions are identified and the signal routing mechanism is explained. The second phase identifies how default paths can be designed.

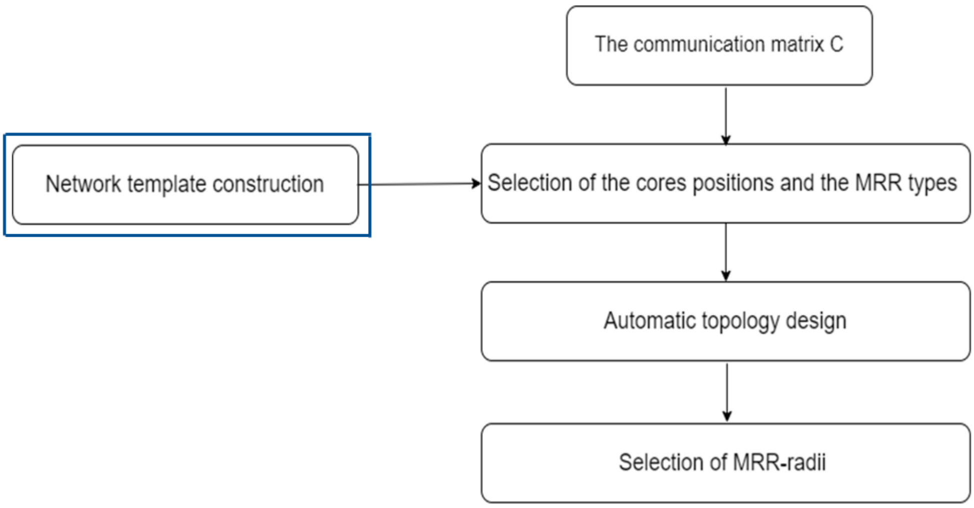

Figure 4.1.: The main steps of designing the topology.

4.2.2.1. Core Positions and MRR Positions

The crossbar structure (Le Beux, Li, Nicolescu, Trajkovic, & O'Connor, 2014) and the ring structure (Le Beux et al., 2011) are the typical WROWoC waveguide structures. The ring structure requires high connections and has poor scalability. Therefore, it consumes much more power and has much higher propagation losses (Ramini, Grani, Bartolini, & Bertozzi, 2013b) (Werner et al., 2017). Thus, this work adopts the crossbar structure.

As mentioned before, Equation 4.2 focuses on minimizing a weighted combination of several individual objectives (N_1 :the number of used m1, N_1 : the number of used m2......). In order to minimize Equation 4.2 while satisfying Equation 4.3 and Property 3.1, the number of used m_1 (N_1) should be increased, because N_1 is multiplied by the lowest weight (one) in Equation 4.2. According to Property 3.1, N_1 also represents the number of paths with one ON MRR and zero OFF MRRs (see Equation 3.3). Since each sender can have at most one path with one ON MRR and zero OFF MRRs, N_1 is smaller or equal to the number of senders (η). Therefore, N_1 should be able to take a value from 0 to η. Furthermore, to minimize Equation 4.2, the number of used m_2 (N_2) should be increased; and so on. Thus, the template topology should support the feature 4.4.

$$0 \leq N_i \leq \eta \quad \forall \quad i \in \{1, 2, .., \eta\} \tag{4.4}$$

In the proposed template network, the positions of the cores (a core consists of a sender-receiver pair) are set along the diagonal like in the template topology in (Chen et al., 2023) (Figure 3.2). Figure 4.2 presents an example of the network template for a 5 $\times$ 5 communication matrix (without default paths). As can be noted, the topology supports self-connectivity. In the network template, η rows and 2$\times\eta$-1 columns are designed (In Figure 4.2, the rows are colored green). The first η columns are <u>the main columns</u> and the last η-1 columns are <u>the added columns</u> (In Figure 4.2, the main columns are colored red and the added columns are colored blue).

In the template topology, the x^{th} row has index x-1, where the x^{th} row is the x^{th} green line counted from the top of the template. For example, the η^{th} row is row $\eta - 1$ and the first row is row 0. In addition, the x^{th} column has index x-1, where the x^{th} main / added column is the x^{th} red / blue line counted from the left of the template. For example, the η^{th} main column is the main column $\eta - 1$ and the first added column is the added column 0.

The possible positions of the MRRs are designed as small circles at the intersection of the rows and columns, as can be seen in Figure 4.2. The bottom of the x^{th} added column ($1 \leq x \leq \eta - 1$) is connected to the top of the x^{th} main column. Each core is designed as a number in a circle (After selecting the core positions, the id of the cores will be written in the blue circles in the template topology in Figure 4.2). Each core consists of a sender and a receiver, as can be seen in Figure 4.3. The core emits optical signals into a horizontal waveguide

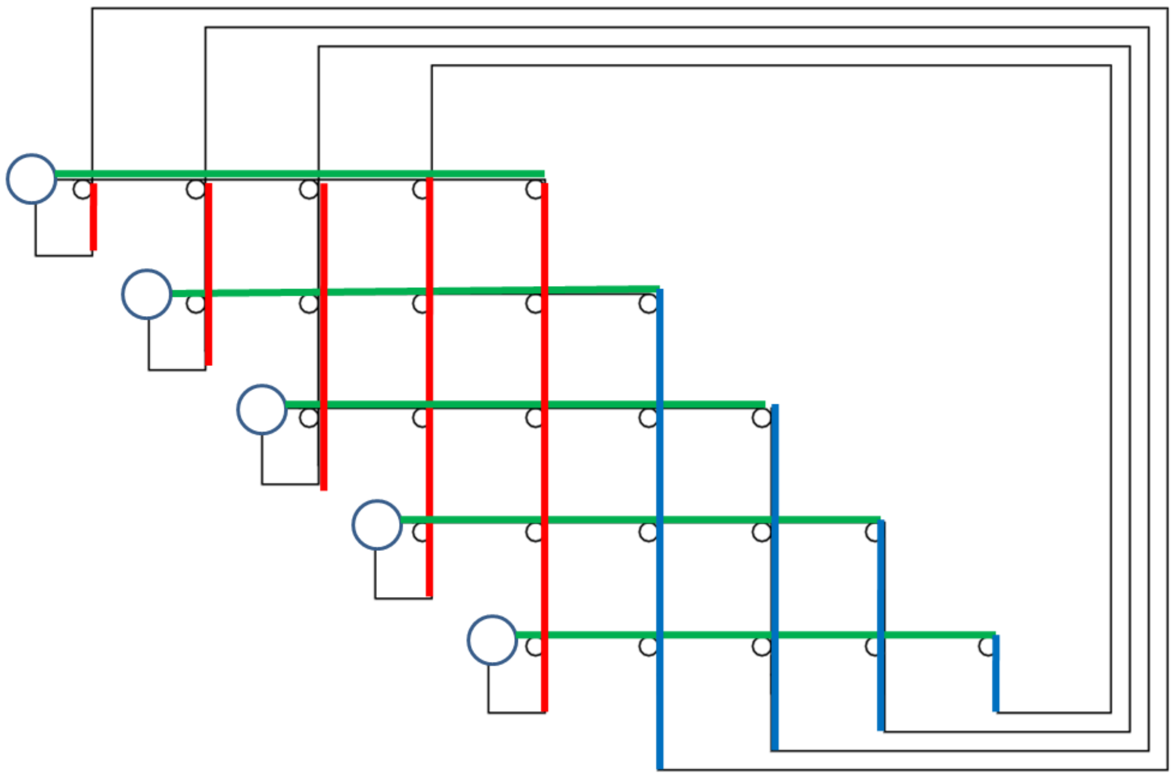

Figure 4.2.: The network template without default paths.

and receives signals from a vertical waveguide. Note that the x^{th} core position has index x-1, where the x^{th} core position is the x^{th} big blue circle counted from the top-left of the template. For example, the first core position is the core position 0, and the second core position is the core position 1.

Note that for a full communication matrix, $N_i = \eta \quad \forall \quad i \in \{1, 2, .., \eta\}$, which indicates that the topology fulfills feature 4.4 and suitable for improving the aggregate parallelism because N_i could be maximized.

4.2.2.2. The Signal Routing Mechanism in the Proposed Topology

Depending on the position of the ON MRR, there are two cases for optical signal transmission:
1) The ON MRR is on a main column (The ON MRR is attached to a main column).

2) The ON MRR is on an added column (The ON MRR is attached to an added column).

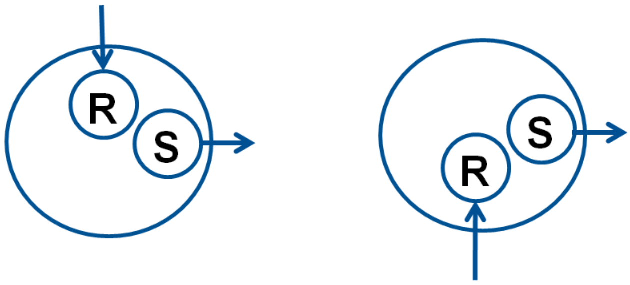

Figure 4.3.: The sender and receiver in each core.

Figure 4.4(a) presents an example of optical signal transmission from the core 2 to the core 4 when the ON MRR is on a main column. As can be seen, if the ON MRR is on a main column, the optical signal is transmitted first from the sender along the horizontal line of this sender until it reaches the ON MRR. Afterwards, the optical signal is coupled to the ring, causing a 90 degree direction change. Next, the optical signal is transmitted along the main column downward until it reaches the receiver.

Figure 4.4(b) presents an example of optical signal transmission from the core 2 (sender 2) to the core 1 (receiver 1) when the ON MRR is on an added column. As can be seen, if the ON MRR is on the x^{th} added column ($1 \leq x \leq \eta - 1$), the optical signal is transmitted first from the sender along the horizontal line connected this sender until it reaches the ON MRR. Afterwards, the optical signal is coupled to the ring, causing a 90 degree direction change. Next, the optical signal is transmitted down the added column to the bottom of this column. Afterwards, since the x^{th} added column is connected to the x^{th} main column , the optical signal is transmitted along the x^{th} main column from the top until it reaches the receiver.

For each path (s, r), the OFF MRRs before the ON MRR are the MRRs between the sender at position s and the ON MRR. The OFF MRRs after the ON MRR are the MRRs between the ON MRR and the receiver at position r.

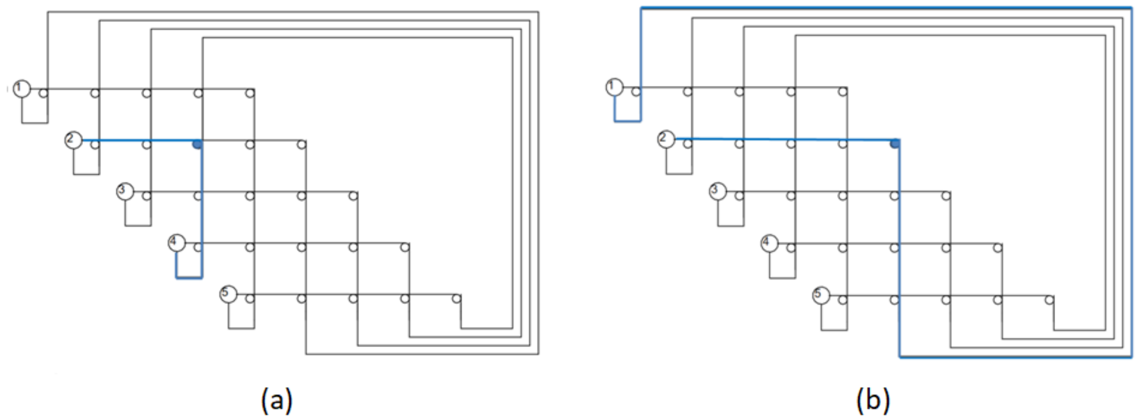

Figure 4.4.: Optical signal transmission.

4.2.2.3. Default Paths

The selection of default paths in the proposed topology is very flexible and does not depend on the positions of the cores. Each signal path (i, j) (the signal path between the core at position i and the core at position j) can be transformed to a default path. There are only two simple constraints for selecting default paths:

- Each core can send signals through at most one default path.
- Each core can receive signals through at most one default path.

The default path from the sender at position i to receiver at position j is designed by connecting the end of the horizontal line connected to the core at position i (row i) to the top of the vertical line connected to the core at position j. Figure 4.5 presents an example of designing a default path. As can be seen in 4.5 (a), to design default paths the blue segments should be connected to the red segments. Figure 4.5 (b) presents the default path from core 5 to core 1. As can be noted the MRR at the intersection of row 4 and the added column 0 is removed.

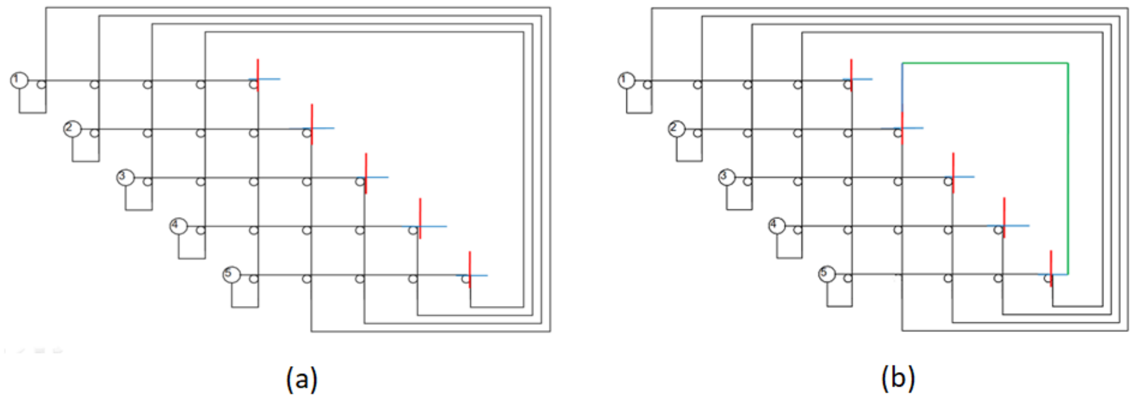

Figure 4.5.: An example of designing a default path.

4.2.3. The Workflow of the Algorithm

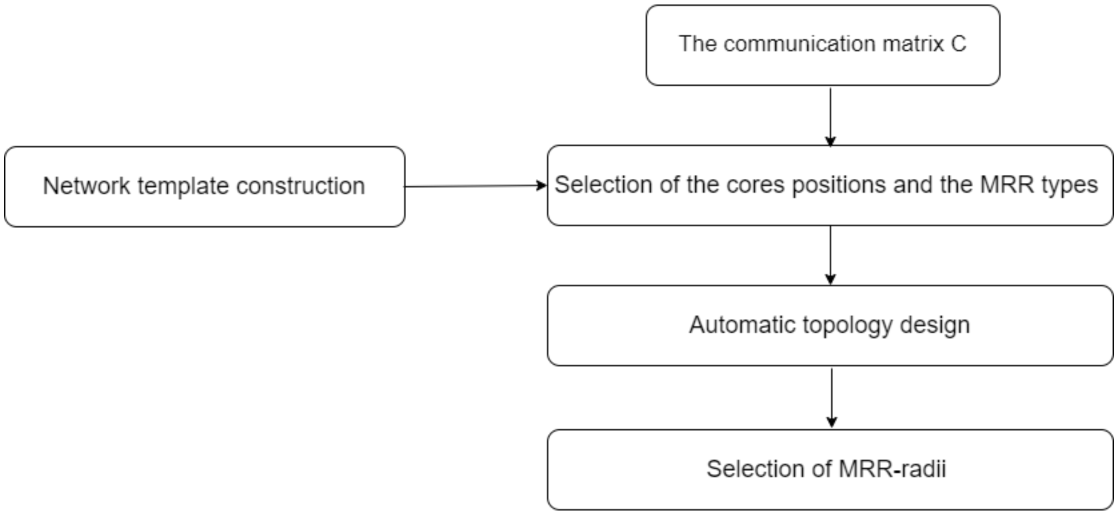

Figure 4.6.: The topology generation process.

This subsection introduces the workflow of the algorithm. Figure 4.6 represents the main steps of the topology generation process. First, according to the communication matrix, the positions of the cores and the MRRs are arranged, and the default paths are selected. ILP model is proposed for this step. In this step, the aim of the algorithm is to minimize Equation 4.2 while satisfying Property 3.2. Next, the topology is designed automatically according to the output of the previous step. The design of the topology is drawn and saved in an image format. In this step, the aim of the algorithm is to reduce the insertion loss. The cores are placed along the diagonal. In addition, the MRRs are drawn and colored with a color according to their type. The differences between the resulting image and the image in Figure 4.2 is that the unused columns and MRRs and the unnecessary waveguide crossing in the resulting image are removed. In addition, the default paths are designed so that the number of waveguide crossing is minimized in order to minimize the insertion loss. Finally, in the last step, a radius is assigned to each MRR type. The algorithm selects the on- and off-resonance MRR types of each path and assigns a radius to each MRR type using the approach presented in (Li et al., 2020).

4.2.4. Re-arrangement of the Cores and Selection of the Default Paths

This section presents the algorithm used to select the core positions, the MRR positions, the MRR types and the default paths. The first subsection introduces the term ***arranging the rows and columns in the same way***. The second subsection presents the relations between the template network and the communication matrix. Finally, the third subsection presents the main idea of the algorithm of the second step in Figure 4.6.

4.2.4.1. Arranging the Rows and Columns in the Same Way

Arranging the rows and columns in the same way means that if the i^{th} row is moved to the j^{th} row position, the i^{th} column is moved to the j^{th} column position. If the i^{th} row is swapped with the j^{th} row, the i^{th} column is also swapped with the j^{th} column. In example Ex.1, the columns and the rows of matrix M are rearranged in the same way. Row r_1 is swapped with row r_2 (r_1 is moved to the third row position and r_2 is moved to the second row position). Similarly, column c_1 is swapped with column c_2.

$$M = \begin{array}{c} \\ r_0 \\ r_1 \\ r_2 \end{array} \begin{array}{c} \begin{array}{ccc} c_0 & c_1 & c_2 \end{array} \\ \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \end{array} \Rightarrow M = \begin{array}{c} \\ r_0 \\ r_2 \\ r_1 \end{array} \begin{array}{c} \begin{array}{ccc} c_0 & c_2 & c_1 \end{array} \\ \begin{pmatrix} 1 & 3 & 2 \\ 7 & 9 & 8 \\ 4 & 6 & 5 \end{pmatrix} \end{array} \quad \text{(Ex.1)}$$

4.2.4.2. The Relations Between the Communication Matrix and the Network Template

Each row index in the communication matrix represents a sender. Each column index in the communication matrix represents a receiver.

Each cell (i, j) equal to 1 in the communication matrix represents connection (i, j). There is a connection between the sender represented by row index i and the receiver represented by column index j, if the entry (i, j) in the communication matrix is equal to 1.

Each sender x is represented by a row at position i (row i) in the communication matrix. To find the receivers that sender x communicates with, the connections in row i are used to identify the receivers of sender x. Each receiver y is represented by a column at position j in the communication matrix. To find the senders that receiver y communicates with, the connections in column j are used to identify the senders that sends data to receiver y. For example, the sender s_2 in communication matrix Ex.2 is represented by row 1. According to the communication matrix, sender s_2 communicates with receivers represented by columns 0 and 2. Columns 0 and 2 represents the connections of receivers r_0 and r_1, respectively. Therefore, sender s_2 communicates with receivers r_0 and r_1.

$$M = \begin{array}{c} \\ s_0 \\ s_2 \\ s_1 \end{array} \begin{array}{c} \begin{matrix} r_0 & r_2 & r_1 \end{matrix} \\ \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \end{array} \qquad \text{(Ex.2)}$$

The connection (i, j) is a ***default connection***, if the signal path from the sender represented by row index i in the communication matrix to the receiver represented by column index j in the communication matrix is a default path.

The main relations between the template topology and the communication matrix:

1. Each connection in the lower triangle (i, j) (i>j) of the communication matrix represents an MRR attached to added column j and the row i, if this connection is not a default connection. Take Figure 4.7 as an example, if $c_{3,1} = 1$ and the connection (3, 1) is not a default connection, there will be an MRR in the red square in the template topology.

2. Each connection in the upper triangle or in the diagonal (i, j) (i $\leq$ j) of the communication

matrix represents an MRR attached to main column j and row i, if this connection is not a default connection. Take Figure 4.7 as an example, if $c_{1,3} = 1$ and the connection (1, 3) is not a default connection, there will be an MRR in the green square in the template topology.

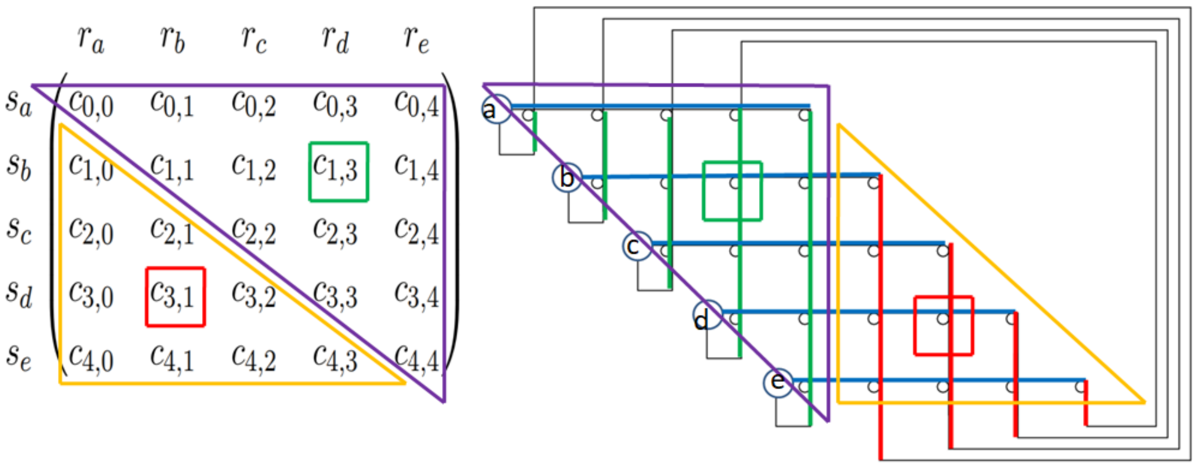

Figure 4.7.: The relations between the communication matrix and the network template

3. Each connection in row i in the communication matrix represents an MRR attached to row i in the template network, if this connection is not a default connection. Each connection (i, j) (i $\leq$ j) in the column i in the communication matrix represents an MRR attached to the main column i, if this connection is not a default connection. The main column i and row i in the Network template are connected to core x placed at core position i in the template topology. Since core x consists of sender x and receiver x, the column i and the row i in the communication matrix represent the connections of the core x. The non-zero cells in row i in the communication matrix identify the receivers that receives data from sender x. The column i in communication matrix identify the senders that sends data to receiver x. In summary, the connections of core x placed at position i in the template network are the connections in column i and the connections in row i in the communication matrix.

4. According to the point 3, to arrange the cores on the template, the columns and the rows and arranged in the same way. After the rearrangement, the id of the sender at the row position i (row i) in the communication matrix represents the id of the core at core position i in the network template. For example, if s_4 is represented by row 2 in the communication matrix, c_4 is placed at the core position 2 in the template topology.

4.2.4.3. The Algorithm

This subsection presents the main idea of the algorithm used to select the core positions, the MRR positions, the MRR types and the default paths. The algorithm iterates over all possible core placement. For each iteration, the rows and the columns in the communication matrix are rearranged according to the placement of the cores. As mentioned in Subsection 4.2.4.2, the rows, the columns and the cores are arranged in the same way. For example, if core x is placed at the first position, the row that represents sender x in the input communication matrix is moved to the first row position in the communication matrix and the column that represents receiver x in the input communication matrix is moved to the first column position in the communication matrix. Then, a type is assigned to each MRR according the new placement and to Property 3.1. Afterward, the default path are selected so that the sum in Equation 4.2 is minimized. Finally, after performing all iterations, the model selects the placement that minimizes Equation 4.2. This algorithm is implemented using ILP.

4.2.4.4. The ILP Model

An ILP model is used to determine the positions of the cores and the positions of the MRRs on the template topology. In addition, this model aims to assign a type to each MRR in the topology. Furthermore, this model is used to determine the default paths. These are the output of the model. The input is the communication matrix.

A. Matrix rearrangement and Default Paths Selection

The set N represents all integer constants from 0 to η - 1 (N = $\{0, 1, ..., \eta - 1\}$). For each $x \in N$ and $y \in N$, a binary variable $R_{x,y}$ is introduced to represent the new position of each row in the matrix. $R_{x,y}$ =1 indicates that the row x in the communication matrix is moved to the row position y. Therefore, constraint C.1.1 is set to ensure that each row in the communication matrix is moved to exactly one new row position. Similarly, constraint C.1.2 is set to ensure that each new row position is selected exactly once (Exactly one row is moved to each new row position).

$$\sum_{y \in N} R_{x,y} = 1, \qquad \forall \quad x \in N. \tag{C.1.1}$$

$$\sum_{x \in N} R_{x,y} = 1, \qquad \forall \quad y \in N. \tag{C.1.2}$$

Default Path Selection:

For each $x \in N$ and $y \in N$, a binary variable $D_{x,y}$ is introduced to represent which paths are default paths. $D_{x,y} = 1$ indicates that connection (x, y) in the communication matrix is a default connection. Therefore, to ensure that each sender has at most one default path, the constraint C.1.3 is set. Similarly, to ensure that each receiver is selected in at most one default path, the constraint C.1.4 is set.

$$\sum_{y \in N} D_{x,y} \leq 1, \qquad \forall \quad x \in N. \tag{C.1.3}$$

$$\sum_{x \in N} D_{x,y} \leq 1, \qquad \forall \quad y \in N. \tag{C.1.4}$$

Communication Matrix rearrangement:

1) Row rearrangement: To perform the row rearrangement, for each $x \in N$ and $y \in N$, a binary variable $B_{x,y}$ is introduced. Each binary variable $B_{x,y}$ is used to represent the value of the cell (x, y) in the new matrix B. To perform this step, the constraints in C.1.5 are set.

$$B_{x,y} \geq C[i][y] - (1 - R_{i,x}), \qquad \forall \quad x \in N, \quad \forall \quad i \in N, \quad \forall \quad y \in N \tag{C.1.5}$$

C[i][y] represents the value of the input communication matrix at position (i, y). Therefore, C[i][y] could be either 0 or 1. The value of C[i][y] is assigned to $B_{x,y}$, if row i is moved to the new row position x ($R_{i,x} = 1$). In this case, $B_{x,y} \geq$ C[i][y] - (1-$R_{i,x}$) = C[i][y]. The optimization objective aims to minimize the sum of the values of the binary variables $B_{x,y}$. Therefore, if C[i][y]=1 and $R_{i,x}$ =1, $B_{x,y}$ is forced to take 1. Otherwise, if C[i][y]=0 and $R_{i,x}$ =1, $B_{x,y}$ can take 1 or 0, and due to the optimization objective, $B_{x,y}$ takes the value 0.

If row i is not moved to the new row position x ($R_{i,x}$ =0) , $B_{x,y} \geq$ C[i][y] - (1-$R_{i,x}$) = C[i][y] - 1 . Since the value of C[x][y] $\leq$ 1, the constraint will not affect the value of $B_{x,y}$.

2) Column rearrangement and default path selection: To perform the column rearrangement,

for each $x \in N$ and $y \in N$, a binary variable $b_{x,y}$ is introduced to represent the value of the (new) matrix b at position (x, y) after performing the row and column rearrangement and selecting the default paths. To perform the column rearrangement and to select the default paths, the constraints in C.1.6 are set. The column rearrangement is done to the matrix B represented by the values of $B_{x,i}$, where $x \in N$ and $i \in N$ ($B_{x,i}$ represents the cell (x, i) in matrix B). As mentioned before, the rows and columns are rearranged in the same way. Therefore, the values of the binary variables $R_{i,y}$ ($i \in N$ and $y \in N$) are also used to perform the column rearrangement. If a connection (x, y) ($x \in N$ and $y \in N$) in the matrix b is a default connection ($D_{x,y} = 1$), 0 is assigned to $b_{x,y}$

$$b_{x,y} \geq B_{x,i} - (1 - R_{i,y}) - D_{x,y}, \qquad \forall \quad y \in N, \forall \quad i \in N, \forall \quad x \in N. \tag{C.1.6}$$

As can be noted, similar to the previous step, if $R_{i,y}$ =0, the constraint will not affect the binary variable $b_{x,y}$. In addition, if $D_{x,y} = 1$, $B_{x,i}$ - (1- $R_{i,y}$) - $D_{x,y}$ = $B_{x,i}$ - (1- $R_{i,y}$)- 1 ≤ 0. Thus, due to optimization target which aims to minimize the sum of the binary variables $b_{x,y}$,if $D_{x,y}$=1, $b_{x,y}$ will take the value 0. If $D_{x,y} = 0$ and $R_{i,y} = 1$, $b_{x,y}$ will take the value $B_{x,i}$.

It is possible to modify the constraint C.1.6 to perform to independent rearrangement of columns and rows (If the i^{th} row is moved to the j^{th} row position, the i^{th} column can move to any column position regardless of row rearrangement). The independent rearrangement of columns and rows provides slightly better aggregate parallelism compared to arranging the rows and columns in the same way. However, the independent rearrangement of columns and rows causes many additional waveguide crossings to bring each sender close to its receiver in the last step. This causes higher insertion loss. Therefore, in order to reduce the total insertion loss, the proposed topology is designed so that the rows and columns are arranged in the same way.

B. Assigning a type to each MRR

For each $i \in N$ and $j \in N$, an integer variable $T_{i,j}$ is introduced. The lower bound of these variables is 0. If a non-zero value is assigned to $T_{i,j}$, the value assigned to $T_{i,j}$ represents the index of the MRR type that the path from the core at position i to the core at position j activates. For example, if $T_{i,j} = 2$, m_2 is the type of the ON MRR when sending data from the core at position i to the core at position j.

The matrix b represents the MRRs in the template topology ($b_{i,j}$ represents cell (i, j)). For example, if $b_{i,j} = 1$ (i < j), there is an MRR at the intersection of the row i and the main column j in the template topology. Thus, if 0 is assigned to $b_{i,j}$, 0 is assigned to $T_{i,j}$. Otherwise, if 1 is assigned to $b_{i,j}$, an integer variable larger than 0 is assigned to $T_{i,j}$. As mentioned before, the values in the upper triangle and in the diagonal of the matrix $b_{i,j}$ ($i \in N$ and $j \in N$, $i \leq j$) represent the connections on the main columns, while the values in the lower triangle of the matrix $b_{i,j}$ ($i \in N$ and $j \in N$, $i > j$) represent the connections on the added columns. As mentioned before, according to Property 3.1, for each path p, if $m_x = ON_p$ (x is an integer variable), then $OFF_p \subseteq \{m_1, m_2, ..., m_{x-1}\}$. For each path (i, j), the set $X_{(i,j)}$ represents the paths that activate the off-resonance MRRs in path (i, j). Therefore, for each path (i, j) other than the default paths, the value assigned to $T_{i,j}$ should be larger than the value assigned to $T_{u,v}$, where (u,v) in $X_{(i,j)}$. It is possible to determine the set $X_{(i,j)}$ for each path from the communication matrix depending on the position of the ON MRR, as mentioned in subsections 4.2.2.2 and 4.2.4.2. Therefore, the constraints in C.1.7, C.1.8 and C.1.9 are introduced. The set U represents the set of the integer pairs (i, j), where i < j and j < η. L represents the set of the integer pairs (i, j) where i > j and i < η. To ensure that the value of $T_{i,j}$ is 0, if there is no connection in $b_{i,j}$ ($b_{i,j}$ =0), - M (1-$b_{i,j}$) is set in each constraint. If $b_{i,j}$=0, - M (1-$b_{i,j}$) = -M, therefore, the constraint is inactive and $T_{i,j}$ takes the lowest possible value 0 due to the optimization target which aims to minimize the sum of the variables $T_{i,j}$, $i \in N$ and $j \in N$.

$$T_{i,i} \geq 1 - M * (1 - b_{i,i}), \qquad \forall \quad i \in N, \tag{C.1.7}$$

$$\forall \quad (i,j) \in U,$$

$$\forall x \in \{0, 1, .., j - i - 1\} : \qquad T_{i,j} \geq 1 + T_{i,i+x} - M * (1 - b_{i,j})$$

$$\forall x \in \{0, 1, .., j{-}i{-}1\} : \qquad T_{i,i} \geq 1{+}T_{x+1+i,j}{-}M{*}(1{-}b_{i,j}) \tag{C.1.8}$$

$$\forall \quad (i,j) \in L,$$

$$\forall x \in \{0, 1, .., \eta - i - 1\} : \qquad T_{i,j} \geq 1 + T_{i,x+i} - M * (1 - b_{i,j})$$

$$\forall x \in \{0, 1, .., j - 1\} : \qquad T_{i,j} \geq 1 + T_{i,x} - M * (1 - b_{i,j})$$

$$\forall x \in \{0, 1, .., j\} : \qquad T_{i,j} \geq 1 + T_{x,j} - M * (1 - b_{i,j})$$

$$\forall x \in \{0, 1, .., \eta{-}2{-}i\} : \qquad T_{i,j} \geq 1{+}T_{x+1+i,j}{-}M{*}(1{-}b_{i,j}) \tag{C.1.9}$$

Constraint C.1.7 ensures that type m_1 is assigned to each connection (i, i) in the matrix

b. The constraints in C.1.8 are set to assign a type to each MRR in a main column in the template topology. For each path (i, j) that activates an MRR in a main column in the template topology, the constraint in C.1.8 ensure that the value assigned to $T_{i,j}$ is larger than the values assigned to $T_{u,v}$, where (u,v) in $X_{(i,j)}$. For each path (i, j) that activates an MRR in a main column in the template topology, the first constraint in C.1.8 ensures that the value assigned to $T_{i,j}$ (The index of the ON MRR type) is larger than the indexes of the types assigned to the OFF MRRs before the ON MRR in path (i, j). For each path (i, j) that activates an MRR in a main column in the template topology, the second constraint in C.1.8 ensures that the value assigned to $T_{i,j}$ (The index of the MRR type) is larger than the indexes of the types assigned to the OFF MRRs after the ON MRR in path (i, j). Take Figure 4.8 as an example (The cell (i, j) in matrix T represents the values assigned to $T_{i,j}$), to assign a value to $T_{1,3}$ (the cell in the red square), the first constraint in C.1.8 ensures that the value assigned to $T_{1,3}$ is larger that the values assigned to the cells in the green square, if $b_{1,3} = 1$. In addition, if $b_{1,3} = 1$, the second constraint in C.1.8 ensures that the value assigned to $T_{1,3}$ is larger that the values assigned to the cells in the blue square.

$$
\begin{array}{c}
 \\ s_a \\ s_b \\ s_c \\ s_d \\ s_e
\end{array}
\begin{array}{c}
\begin{array}{ccccc} r_a & r_b & r_c & r_d & r_e \end{array} \\
\begin{pmatrix}
T_{0,0} & T_{0,1} & T_{0,2} & T_{0,3} & T_{0,4} \\
T_{1,0} & T_{1,1} & T_{1,2} & T_{1,3} & T_{1,4} \\
T_{2,0} & T_{2,1} & T_{2,2} & T_{2,3} & T_{2,4} \\
T_{3,0} & T_{3,1} & T_{3,2} & T_{3,3} & T_{3,4} \\
T_{4,0} & T_{4,1} & T_{4,2} & T_{4,3} & T_{4,4}
\end{pmatrix}
\end{array}
$$

Figure 4.8.: The 5x5 matrix T

The constraints in C.1.9 are set to assign a type to each MRR in an added column in the template topology. For each path (i, j) that activates an MRR in an added column in the template topology, the constraints in C.1.9 ensure the value assigned to $T_{i,j}$ is larger than the values assigned to $T_{u,v}$, where (u,v) in $X_{(i,j)}$. For each path (i, j) that activates an MRR in an added column in the template topology, the first and the second constraints in C.1.9 ensure that the value assigned to $T_{i,j}$ (The index of the ON MRR type) is larger than the indexes of the types assigned to the OFF MRRs before the ON MRR in path (i, j). For each path (i, j) that activates an MRR in an added column in the template topology, the third and the fourth constraints in C.1.9 ensure that the value assigned to $T_{i,j}$ (The index of the MRR type) is larger than the indexes of the types assigned to the OFF MRRs after the ON MRR in path (i, j). Take Figure 4.9 as an example, if $b_{3,1} = 1$, to assign a value to $T_{3,1}$ (the cell in the red square), the first and second constraints in C.1.9 ensure that the value assigned to $T_{3,1}$ is larger that the values assigned to the cells in the green and the yellow squares, respectively. In addition, if $b_{3,1} = 1$, the third and fourth constraints in C.1.9 ensure that the value assigned to $T_{3,1}$ is larger that the values assigned to the cells in the blue and the black squares, respectively.

$$\begin{array}{c} \\ s_a \\ s_b \\ s_c \\ s_d \\ s_e \\ s_f \end{array} \begin{array}{c} \begin{array}{cccccc} r_a & r_b & r_c & r_d & r_e & r_f \end{array} \\ \begin{pmatrix} T_{0,0} & T_{0,1} & T_{0,2} & T_{0,3} & T_{0,4} & T_{0,5} \\ T_{1,0} & T_{1,1} & T_{1,2} & T_{1,3} & T_{1,4} & T_{1,5} \\ T_{2,0} & T_{2,1} & T_{2,2} & T_{2,3} & T_{2,4} & T_{2,5} \\ T_{3,0} & T_{3,1} & T_{3,2} & T_{3,3} & T_{3,4} & T_{3,5} \\ T_{4,0} & T_{4,1} & T_{4,2} & T_{4,3} & T_{4,4} & T_{4,5} \\ T_{5,0} & T_{5,1} & T_{5,2} & T_{5,3} & T_{5,4} & T_{5,5} \end{pmatrix} \end{array}$$

Figure 4.9.: The 6x6 matrix T

The optimization target is to minimize the sum of $T_{i,j}$. Since $T_{i,j}$ depends on the values of $b_{i,j}$, minimizing the sum of $T_{i,j}$ leads to minimizing the sum of $b_{i,j}$. Similarly, since $b_{i,j}$ depends on the values of $B_{i,j}$ and $D_{i,j}$, minimizing the sum of $T_{i,j}$ leads to minimizing the sum of $B_{i,j}$ and maximizing the sum of $D_{i,j}$.

Minimize $\sum_{i\in N, j\in N} T_{i,j}$

Subject to: C.1.1, C.1.2, C.1.3, C.1.4, C.1.5, C.1.6, C.1.7. C.1.8 and C.1.9.

4.2.4.5. Maximizing Only the Worst Case Parallelism

The main target of this project is to improve the worst case parallelism and the aggregate parallelism at the same time. This subsection presents an approach that aims only to maximize the worst case parallelism by modifying the optimization target of the proposed ILP in Subsection 4.2.4.4.

An integer variable $\boldsymbol{ma}$ is introduce to represent the maximum value of variables $T_{i,j}$, $i \in N$ and $i \in N$. Therefore the constraint C.1.10 is set. The minimum number of MRR types in a topology ($\boldsymbol{low}$) can be calculated using formula 4.4 (Chen et al., 2023). Therefore, the lower bound of the variable $\boldsymbol{ma}$ is set to $\boldsymbol{low}$. Setting this lower bound significantly reduces the ***runtime*** of the ILP model.

$$T_{i,i} \leq ma, \qquad \forall \quad i \in N, \quad \forall \quad j \in N, \tag{C.1.10}$$

$$Low = \max(\max_i \sum_{j \in N} C[i][j], \max_j \sum_{i \in N} C[i][j]) - 1 \tag{4.4}$$

In addition, the number of used MRR should be minimized. For each $i \in N$ and $j \in N$, an integer variable $t_{i,j}$ is introduced to represent whether $T_{i,j}$ takes a non zero value. Therefore, the constraint C.1.11 is set. If $T_{i,j} = 0$, $t_{i,j}$ can take 1 or 0. Due to the optimization objective which aims to minimize the sum of the variables $t_{i,j}$, $i \in N$ and $j \in N$, 0 is assigned to $t_{i,j}$, if $T_{i,j} = 0$.

$$T_{i,j} \leq M * t_{i,j} \qquad \forall \quad i \in N, \quad \forall \quad j \in N, \tag{C.1.11}$$

In this model, the first optimization target is to minimize $\boldsymbol{ma}$. The second optimization target is to minimize the sum of variable $t_{i,j}$ ($i \in N$ and $j \in N$) (The number of used MRR). Therefore, the optimization target can be formulated as follows (because $\eta^2 \geq \sum_{i \in N, j \in N} t_{i,j}$):

Minimize $\eta^2 \times ma + \sum_{i \in N, j \in N} t_{i,j}$

Subject to: C.1.1, C.1.2, C.1.3, C.1.4, C.1.5, C.1.6, C.1.7. C.1.8, C.1.9, C.1.10 and C.1.11

The ILP model in Subsection 4.2.4.4 optimizes the aggregate parallelism and the worst case parallelism at the same time. The ILP model presented in this subsection optimizes only the worst case parallelism. This ILP model can terminate much faster when the size of the communication matrix is large ($\eta > 9$). In this master's thesis, ***the main ILP model is the ILP model presented in Subsection 4.2.4.4***, which is used in the comparison results in Chapter 5.

4.2.5. The Output of the ILP Model

The output values of the binary variables $R_{i,j}$ identify the new position of each core in the topology, as mentioned in Subsection 4.2.4.2. The program stores the new position of each core in the list $rank\_s$.

The output values of the binary variables $D_{i,j}$ identify the default paths. Then, if $D_{i,j} = 1$, the program checks whether there is a connection between the core at position i and the core at position j, in which case, the algorithm stores the pair (i,j) in the list $default\_paths$.

Depending on the values of $T_{i,j}$, where i $\in$ N and j $\in$ N, the matrices L and U are generated, as can be seen in M1 and M2. The size of the two matrices is $\eta \times \eta$. The matrix U is then concatenated with the matrix L to construct the matrix T2, as can be seen in 4.2. The size of the resulting matrix is $\eta \times 2 * \eta$.

$$L[i][j] = \begin{cases} T_{i,j}, & i > j \\ 0, & Otherwise \end{cases} \tag{M1}$$

$$U[i][j] = \begin{cases} T_{i,j}, & i \leq j \\ 0, & Otherwise \end{cases} \tag{M2}$$

$$T2 = \begin{bmatrix} U[0,0] & \dots & U[0,\eta-1] & L[0,0] & \dots & L[0,\eta-1] \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ U[\eta-1,0] & \dots & U[\eta-1,\eta-1] & L[\eta-1,0] & \dots & L[\eta-1,\eta-1] \end{bmatrix} \tag{4.2}$$

4.2.6. Drawing the Topology

After selecting the cores' position, the default paths, and the position and the type of each MRR in the topology, the following sections present how the topology is designed and drawn automatically.

4.2.6.1. Drawing the MRR

The drawing process of the MRRs depends on the output of the previous step. The type of each used MRR is stored in matrix T2. The program stores the index of each new added columns, that has at least one connection, in a list New. In the example in matrix 4.5, the added columns, that have at least one value different from zero, are the first added column (column η) as well as the second added column (column η+1). Therefore, the list is New = [0, 1]. The length of the list New is w. Next, the program removes the added columns that are not used. Then, the program iterates over the rows and the columns of matrix T2. If the value of T2[i, j] is larger than 0, the program first draws a circle at the position (an_x + d*i,an_y + d*j), where an_x, an_y, and d are constants. The distance between the columns and between the rows (the value of constant d) depends on the number of the cores. The more cores there are in the topology, the larger constant d should be. The point (an_x, an_y) represents the start point (If T2[0, 0] $>$ 0, there will be an MRR in position (an_x, an_y). The position of the topology drawing in the image depends on the values of point (an_x, an_y). Changing the values of point (an_x, an_y) changes the position of the topology drawing in the image. For example, if the value of an_x increases, the topology drawing will be shifted to the right. Afterwards, the program colors each circle with a color which depends on the value of T2[i, j]. Each color indicates an MRR type. Figure 4.10 presents an example of drawing the MRRs based on the matrix in 4.5.

$$T2 = \begin{bmatrix} 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 \end{bmatrix} \tag{4.5}$$

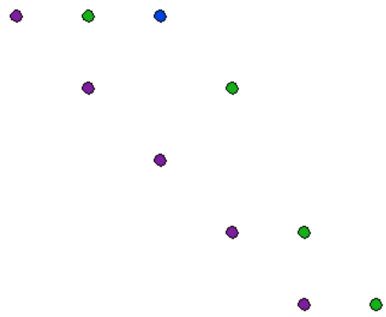

Figure 4.10.: An example of drawing the MRRs

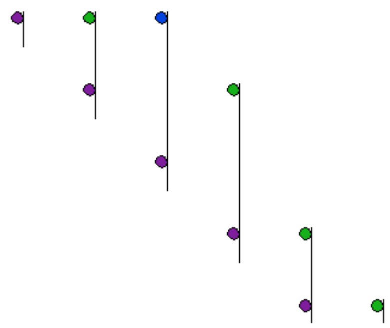

Figure 4.11.: An example of drawing the vertical lines

4.2.6.2. Drawing the Vertical lines

After drawing the MRRs, the vertical lines are drawn. First, the program draws a vertical tangent to each MRR. Then, the program iterates over the main columns of the matrix T2 (from column 0 to column $\eta - 1$). The algorithm connects the MRRs with the same x coordinate, as can be seen in Figure 4.11. The program stores the coordinates (x, y) of the top of each line in the list TOP. In addition, the program stores the coordinates of the bottom of each line in the list Slaves. If a main column i has no connection, the lists TOP and Slaves store the values (an_x + 7 + d*i, an_y + d*i). Therefore, the length of the lists TOP and Slaves is always η.

Similarly, the program iterates over the added columns of the matrix T2. First, the program draws a vertical tangent to each MRR. Then, the algorithm connects the MRRs with the same x coordinate, as can be seen in Figure 4.11. The program stores the coordinates of the top of each line in the list New_connect. In addition, the program stores the coordinates of the bottom of each line in the list Bottom. These lists will be used in the next steps. The length of the lists New_connect and Bottom is equal to the length of the list New.

4.2.6.3. Drawing the Horizontal Lines

After drawing the vertical lines, the program first draws a horizontal tangent to each MRR. Then, the program iterates over the rows of the matrix T2. The algorithm connects the MRRs with the same y coordinate, as can be seen in Figure 4.12. The program stores the coordinates of the end of each line in list End. In addition, the program stores the coordinates of the beginning of each line in list Masters. If a row i has no connection, the lists Masters and end store the values (an_x + d*i, an_y + d*i). Therefore, the length of the lists Masters and end is always equal to η.

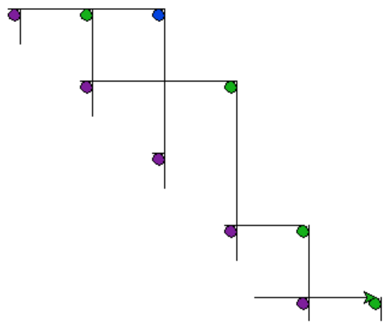

Figure 4.12.: An example of drawing the horizontal lines

4.2.6.4. Connecting the Added Columns

The algorithm connects each added vertical line to the corresponding main vertical line, as can be seen in Figure 4.13. The program ensures that the added lines do not cause additional waveguide crossings with each other. First, the algorithm selects the lowest y-coordinate in the elements in the list Bottom and stores it in a variable y_{max}. Then, the algorithm iterates over all added column i and draws a vertical line from the bottom of each added columns (Bottom[i][0], Bottom[i][1]) until the point (Bottom[i][0], y_{max}). Afterwards, the algorithm iterates over all added columns. For each added column i, the program draws a line with the length (η - i)*10 downwards (↓) starting from the point (Buttom[i][0], y_{max}). Then, the program draws a line with length d*(w-i) + (η - i)*10 to the right ($\rightarrow$) starting from the previous position. Afterwards, the algorithm draws a line with length d*η + (η - i)*20 upwards ($\uparrow$) starting from the previous position. Next, the program draws a horizontal line to the left ($\leftarrow$) starting from the previous position to reach the point with x coordinates TOP[New[i]][0]. Finally, the program draws a horizontal line downwards ($\downarrow$) starting from the previous position to reach the point (TOP[New[i]][0], TOP[New[i]][1]).

All added lines in this step are on the right side. On the left side, the cores will be placed. In this case, accessing the cores does not cause additional waveguide crossings with the waveguide of the topology.

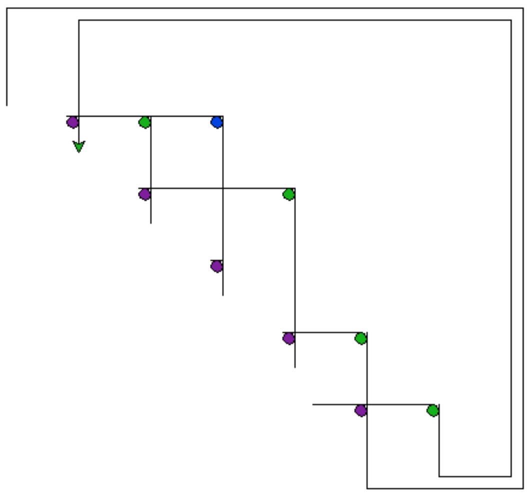

Figure 4.13.: An example of connecting the added columns to the main columns

4.2.6.5. Drawing the Default Paths and the Cores

The list default paths contains pairs, where the first element represents the sender (the index of the vertical line) and the second element represents the receiver. The algorithm iterates over the default paths list. For each default path [i, j], the program connects the end of the horizontal line i (End[i]) to the vertical line j or j + η. If the added columns j exists (if the number j is in the list New), the program connects the end of the horizontal line i (End[i]) to the top of the added column j. Here, the list New_connect is used to identify the top of this vertical line. If the added columns j does not exist the program connects the end of the horizontal line i (End[i]) to the top of the vertical line j (TOP[j]). The algorithm connects the default paths so that the waveguide crossing is minimized, as can be seen in Figure 4.14.

For each iteration, the program compares the x_i coordinate (End[i][0]) and y_i coordinate (End[i][1]) of the end of the horizontal line i with the x_j coordinate (TOP[j][0] or New_connectTop[j][0]) and y_j coordinate (TOP[j][1] or New_connectTop[j][1]) of the vertical line connected to receiver j, respectively. If j in the list New , New_connectTop[j][0] and New_connectTop[j][1] are used. Otherwise, TOP[j][0] and TOP[j][1] are used. Table 4.1 represents the steps that the program performs to draw the default paths depending on the position of the start point (x_i, y_i) and the target point (x_j, y_j), where $| x |$=|x$_i$ - x_j|, $| y |$=|y$_i$ - y_j| and l = (i + 1)*5. u = (η + w - j)*5, if j not in the list New , otherwise, u = (w - j)*5 (w is the length of the list New). The end point of a movement (step) is the starting point of the next movement. The arrow represents the direction in which the program draws the line. The value next to the arrow indicates the length of the line.

For example, if $x_i \leq x_j$ and $y_i \geq y_j$, the program draws a line to the right ($\rightarrow$) from the point (x_i, y_i) with the length $| x |$ to reach the position ($x_i + | x |$, y_i). Then, the program draws a line with the length $| y |$ downwards ($\downarrow$) starting from the previous position ($x_i + | x |$, y_i) to reach the point (x_j, y_j). The default path from core at position 0 to the core at position 4 in Figure 4.14 represents this case.

Case	First move	Second move	Third move	Fourth move
1) $x_i \geq x_j$ and $y_i \geq y_j$	$\rightarrow l$	$\downarrow 17$	$\leftarrow\| x \| + l$	$\downarrow\| y \| - 17$
2) $x_i \leq x_j$ and $y_i \geq y_j$	$\rightarrow\| x \|$	$\downarrow\| y \|$		
3) $x_i \geq x_j$ and $y_i \leq y_j$	$\rightarrow l$	$\uparrow\| y \| + u$	$\leftarrow\| x \| + l$	$\downarrow u$
4) $x_i \leq x_j$ and $y_i \leq y_j$	$\rightarrow\| x \| - 17$	$\uparrow\| y \| + 5$	$\rightarrow 17$	$\downarrow 5$

Table 4.1.: Drawing the default paths for all cases.

Finally, the program connects each master to the corresponding slave (The slave with the same id). The program uses the lists Slaves and Masters for the connection. In addition, the algorithm uses the list *rank_s* to write the id of each core. Figure 4.14 presents an example of drawing the default paths and the cores.

The values l = (i+1)*5, and u = (η +w -j)*5 depend on i and j, respectively, in order to reduce the waveguide crossings. In the example in Figure 4.14, the default path [4, 2] from the core at position 4 to the core at position 2 (from core id 1 to core id 3) represents the case 3. In this default path, $u_{[4,2]} = 25$. The default path [1, 3] from the core at position 1 to the core at position 3 represents the case 3. In this default path, $u_{[1,3]} = 20$. The default path [4,2] ($u_{[0,4]}$ = 25) does not intersect with default path [1.3] ($u_{[1,3]}$ = 20) because ($u_{[1,3]} < u_{[4,2]}$). Figure 4.15(a) represent the proposed topology if ($u_{[1,3]} > u_{[4,2]}$). As can be seen, in this case there are two additional waveguide crossings.

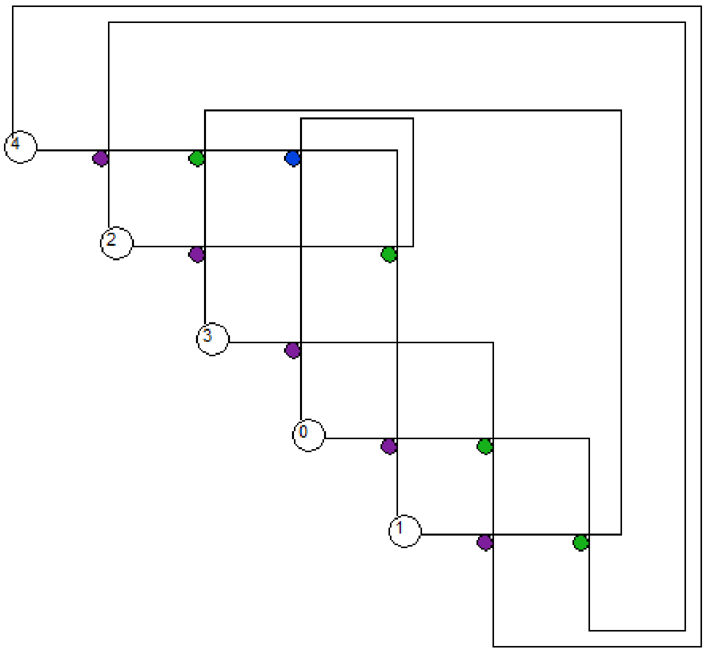

Figure 4.14.: An example of drawing the default paths and the cores

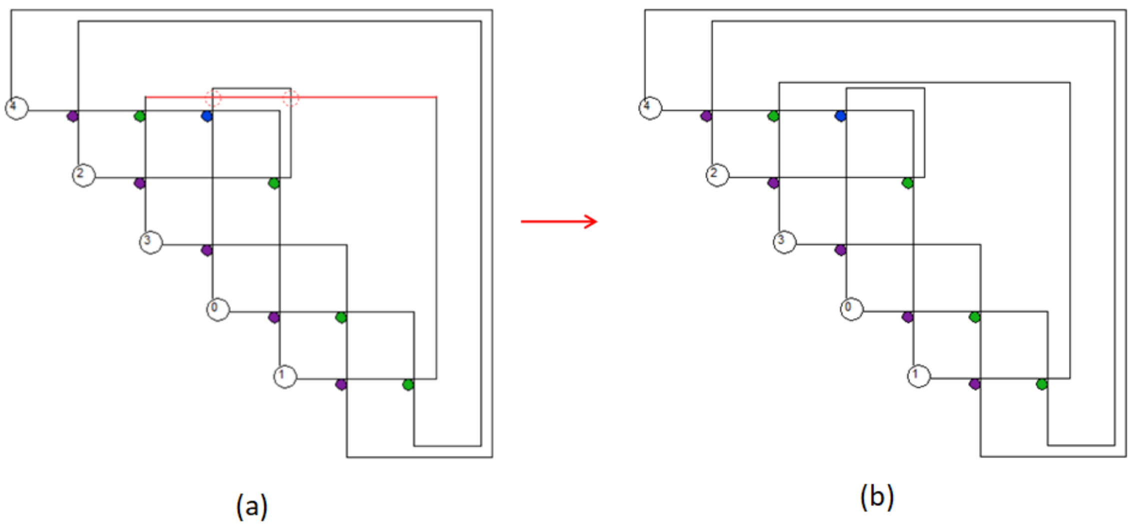

Figure 4.15.: Drawing the default path [4, 2]: (a) $u_{[1,3]} > u_{[4,2]}$ (b) $u_{[1,3]} < u_{[4,2]}$

4.3. Software Environment

4.3.1. The svg-turtle Library

The python library Turtle graphics is used in order to automatically draw the topology. The svg-turtle Library, which supports the Python Turtle interface, is used to draw the topology and then to save the resulting image as an SVG file. The following command is used to install the package (*SaVaGe Turtle*, 2023):

pip install svg_turtle

The size of the SVG file depends on the number of cores in the topology. The more cores there are in the topology, the larger the SVG file is.

4.3.2. PyQt5

PyQt5 is Developed by Riverbank Computing Limited in 04.2016 (Fitzpatrick, 2023). PyQt5 is a python 3 framework (v3.6 or later). PyQt5 allows the designer to create a graphical user interface (GUI) application. The main elements of user interface creation in PyQt5 are Widgets (buttons and checkboxes input fields) (Fitzpatrick, 2023). The following command is used to install the framework PyQt5 (*PyQt5*, 2023):

pip install PyQt5

4.4. Running the Program

The user can execute the proposed program with a graphical user interface. In this case, the framework PyQt5 must be downloaded. The user needs to install gurobipy (GUROBI, 2020). To design a topology, the user should run ***MA_GUI_FARES.py***. Then, the first window will be opened, as can be seen in Figure 4.16. The user should input the number of cores and the other parameter (the safe distance, beta, alpha, etc.). If one of the inputs is not correct, the program will display an error message. Afterwards, the user should click on the button ***Enter the communication matrix***. Next, a second window is opened, as can be seen in Figure 4.17. Then, the user should enter the communication matrix and click on the button ***Generate the topology***. The program designs the topology and outputs an SVG file which contains an image of the topology and a text file which contains the radius of each MRR and the wavelengths assigned to each signal path in the topology.

To input the radius options and the corresponding wavelengths, these variables should be written in the file *mrr_complete.txt*. If this file is in the same folder as the program file, the program will read this file. Otherwise, the program will use the values presented in (Tala & Bertozzi, 2018).

Each line in the file *mrr_complete.txt* should contain one radius option and its corresponding wavelengths as can be seen in the following example, Where the resonant wavelengths of radii 5, 5.25 and 5.5 are presented (The resonant wavelengths of radius 5 are 1496.4 and 1551.3):

```
1 5.0   0.0   0.0 2 1499.1:0.0    1559.3:0.0
2 5.25  0.0   0.0 3 1497.9:0.0    1521.2:0.0    1575.2:0.0
3 5.5   0.0   0.0 3 1511.3:0.0    1578.9:0.0    1583.8:0.0
```

It is possible to run the program without GUI. In this case, the user should run the python file ***MA_Topology_ABID.py***. Then, the user should write the number of cores. Afterwards, the user should enter the rows row by row. In addition, the user should enter the other parameters.

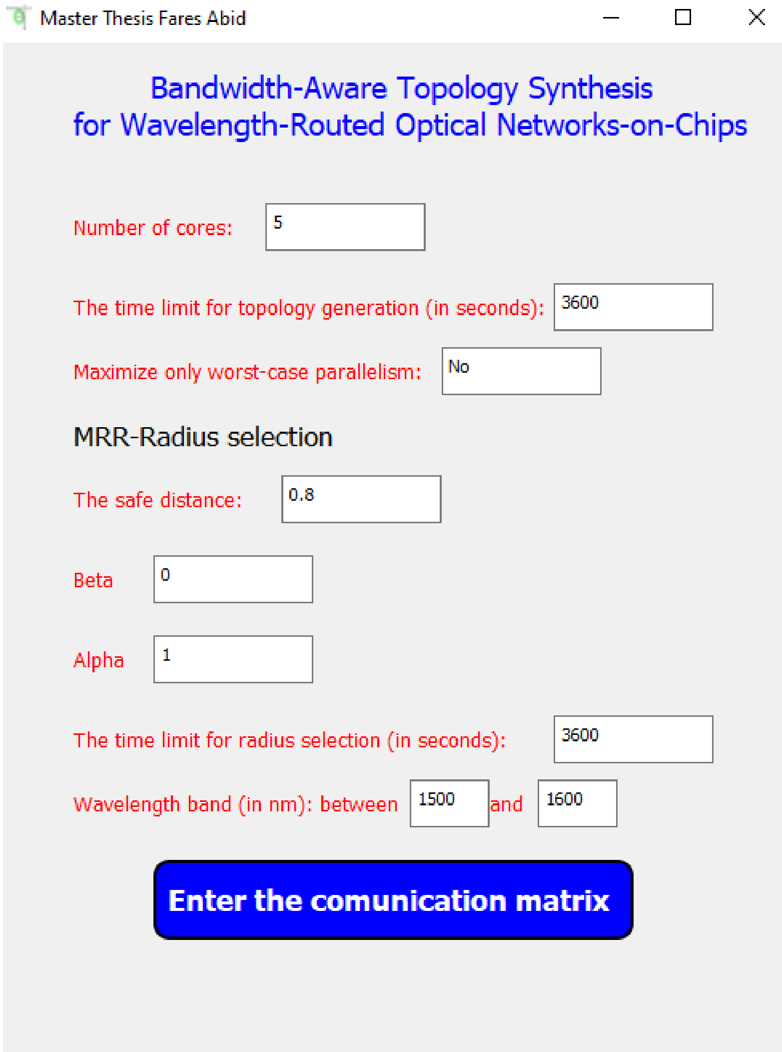

Figure 4.16.: GUI (First window)

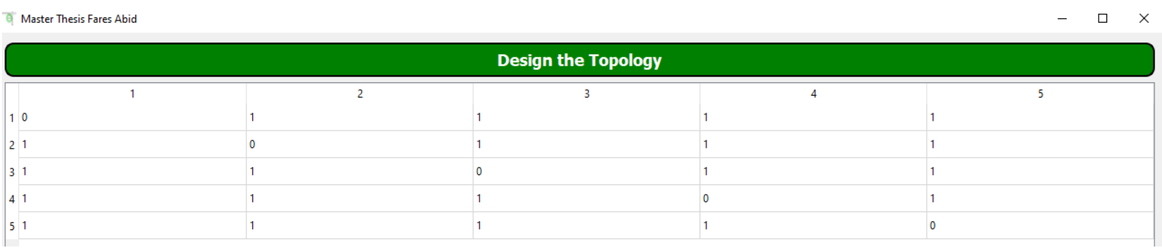

Figure 4.17.: GUI (Second window)

5. Comparison Results and Discussion

This chapter presents and discusses the experimental results for comparing the performance of the proposed topology with the performance of other topologies, such as, Light (Zheng et al., 2021), Snake (Ramini et al., 2013a), and the state-of-the-art parallelism-aware topology (Chen et al., 2023) in terms of improving the aggregate parallelism, the worst case parallelism, the SNR, and in terms of minimizing the power loss and the number of used MRRs. The proposed topology is the first to present an approach that maximizes the aggregate parallelism at the topology generation. Previous research in topology generation includes the topology in (Chen et al., 2023) which aim to maximize the worst case parallelism. Light and Snake topologies are designed for purposes such as reducing the power loss, the number of MRRs and improving the SNR. The first section introduces the terms used to compare the performance of the topologies. The second section presents the parameters used in these experiments. Then, the comparison results are presented and discussed. Different network sizes and communication matrices are used in these comparisons.

5.1. Comparing the Parallelism Performance

The lower the number of OFF MRR types in the path, the more wavelengths can be assigned to this path. In order to compare the performance of two topologies in terms of improving the aggregate parallelism, the average number of OFF MRR types of all paths is calculated using formula 5.1, where P is the set of paths that contain one ON MRR and C represents the set of all connections (C contains the default paths and the set P).

$$P_{average} = \sum_{p \in P} \frac{N_{OFF\_MRR\_in\_p}}{|C|} \tag{5.1}$$

In order to improve the aggregate parallelism, the proposed topology aims to minimize the number of OFF MRR types in each path, while satisfying Property 3.1. In these comparisons, $P_{average}$ is calculated to indicate whether the proposed topology reduces the number of OFF MRR types in the paths of the topology compared to the other topologies.

5.2. The Parameters

This section presents the parameters used in these experiments. The parameters include the MRR-radius options, the available wavelengths, the transmission losses and the safety distance. In these experiments, the suggested parameters in (Tala & Bertozzi, 2018) are used to compare the performance among these topologies. This include 101 MRR-radius options from 5um to 30um with a 0.25um increment step (Tala & Bertozzi, 2018), the available wavelengths between 1500 nm and 1600 nm, and the safety distance is set to 0.8 nm. This is because 0.8 nm is the lowest possible value for receiving crosstalk (quality factor of 10000) less than -20 dB (Preston, Sherwood-Droz, Levy, & Lipson, 2011). The transmission losses values are presented in Table 5.1. The total insertion loss of a path p can be calculated by multiplying the insertion loss of all optical elements in the path p, as mentioned in Section 2.4.

The Parameter	Value	Reference
L_b	-0.005 dB/90°	(Yan et al., 2020)
L_{cr}	-0.04 dB	(Ding, Tang, Liu, Chen, & Sun, 2010)
$L_{pse,on}$	-0.5 dB	(Chan, Hendry, Bergman, & Carloni, 2011)
$L_{pse,off}$	-0.005 dB	(Chan et al., 2011)
$L_{cse,on}$	-0.5 dB	(Xie et al., 2012)
$L_{cse,off}$	-0.04 dB	(Xie et al., 2012)
$C_{p,off}$	-25 dB	(Zheng et al., 2021)
$C_{p,on}$	-25 dB	(Zheng et al., 2021)
C_{Cr}	-40 dB	(Ding et al., 2010)

Table 5.1.: The loss values.

5.3. The Light Topology

The Light topology is an N × (N-1) WRONoC topology which uses parallel switch elements. For a full N × (N-1) communication matrix, the number of used MRRs is reduced by approximately 50 % compared to state-of-the-art topologies, such as the GWOR topology. Figure 5.2 presents the 8*7 Light topology. As can be seen, 24 MRRs are used. The light topology performs better than the λ-route topology and the GWOR topology in terms of reducing the average insertion loss value as well as improving the average SNR value. The Light topology is designed mainly for a full connectivity N × (N-1) communication matrix, but it can also support customized connectivity by removing the redundant MRRs and waveguides.

The Light topology is compared with the proposed topology in terms of reducing the number of MRR types, reducing the average insertion loss (average IL), reducing the worst case insertion loss (Max IL), and improving the aggregate parallelism. The comparison is done for the full connectivity 8 × 7 communication matrix 5.1, where the Light topology performs very well in all aspects.

$$C = \begin{array}{c} \\ s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{array} \begin{array}{c} \begin{array}{cccccccc} r_0 & r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array} \tag{5.1}$$

The proposed topology for this communication matrix is presented in Figure 5.2. As can be seen, the number of MRRs used in Light topology is less than that of the proposed topology. However, the proposed topology uses only 6 different MRR types, where the Light topology uses 8 different MRR types. The worst case insertion loss in the Light topology happens when data is sent from m_1 to s_2. In this path, the optical signal is transmitted through 12 crossing waveguides and 10 OFF PSEs and one ON PSE. Therefore, when the optical signal is transmitted from m_1 to s_2, the total insertion loss is $L_{m_1 s_2}$ = 0.5 + 12 * 0.04 +10* 0.005 = 1.03 dB. The worst case insertion loss in the proposed topology happens when the data is sent from m_2 to s_6. In this path, the optical signal is transmitted through 10 OFF CSEs and one ON CSE. Therefore, when the optical signal is transmitted from m_2 to s_6, the total insertion loss is $L_{m_2 s_6}$ = 0.5 + 10 * 0.04 = 0.9 dB. Thus, the proposed topology performs better than the Light topology in terms of reducing the worst case insertion loss.

Since the paths from each sender in the proposed topology presented in 5.2 are similar, the average insertion loss is calculated in Equation 5.4

$$L_{average} = 8 * (L_{m_2 s_1} + L_{m_2 s_7} + L_{m_2 s_0} + L_{m_2 s_4} + L_{m_2 s_3} + L_{m_2 s_1} + L_{m_2 s_6} + L_{m_2 s_5})/7/8$$

= ((1 ON CSE) + (1 ON CSE + 2 OFF CSE) + (1 ON CSE + 4 OFF CSE) + (1 ON CSE + 6 OFF CSE) + (1 ON CSE + 8 OFF CSE) + (1 ON CSE + 10 OFF CSE) + (12 OFF CSE))/7 =

(0.5 + (0.5 + 2*0.04) + (0.5 + 4*0.04) + (0.5 + 6*0.04) + (0.5 + 8*0.04) + (0.5 + 10*0.04) + (12*0.04))/7 = 0.668 dB (5.4)

The total insertion loss in the proposed topology is 37.4 dB. For the Light topology, Table 5.2 presents the number of ON PSE , the number of OFF PSEs and the number of crossing waveguides for each path (presented as #ON PSE, #OFF PSE, #crossing waveguides in each cell). As can be seen, the total insertion loss is 38.64 dB and the average insertion loss is 0.69 dB, which is higher than that of the proposed topology. Therefore, the proposed topology performs better than the Light topology in terms of minimizing the average insertion loss and the total insertion loss. Note that the Light topology has more waveguide bends than the proposed topology. Therefore, if the waveguide bends were considered in the calculation of the power loss, the difference in total insertion loss between the two topologies would be larger.

	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	Average IL
s_1	×	1,10,12	1,8,10	1,6,8	0,6,6	1,4,4	1,2,2	1,0,0	0.69 dB
s_2	1,8,8	×	1,6,8	1,4,6	1,6,8	0,6,6	1,4,4	1,2,2	0.69 dB
s_3	1,6,6	1,4,4	×	1,2,4	1,8,10	1,6,8	0,6,6	1,4,4	0.69 dB
s_4	1,4,4	1,2,2	1,0,0	×	1,10,12	1,8,10	1,6,8	0,6,6	0.69 dB
s_5	0,6,6	1,4,4	1,6,6	1,8,8	×	1,2,4	1,4,6	1,6,8	0.69 dB
s_6	1,6,8	0,6,6	1,4,4	1,6,6	1,0,0	×	1,6,8	1,8,10	0.69 dB
s_7	1,4,6	1,6,8	0,6,6	1,4,4	1,2,2	1,4,4	×	1,10,12	0.69 dB
s_8	1,2,4	1,4,6	1,6,8	0,6,6	1,4,4	1,6,6	1,8,8	×	0.69 dB

Table 5.2.: The optical elements in each path in the Light topology.

Table 5.3 represents a comparison between the Light topology and the proposed topology. As can be seen, for the Light topology $P_{average} = 186/56 = 3.32$ OFF MRR types/Path. After running the method proposed in (Li et al., 2020) (alpha=1 and beta=0), $v_{worst} = 4$. The proposed topology is presented in Figure 5.2. For the proposed topology $P_{average} = 120/40 = 2.14$ OFF MRR types/Path. After running the method proposed in (Li et al., 2020) (alpha=1 and beta=0), the radii 12.25, 10.75, 29, 28, 15.25, 27.5, 8.5 and 28.25 are assigned to r_1, r_2, r_3,

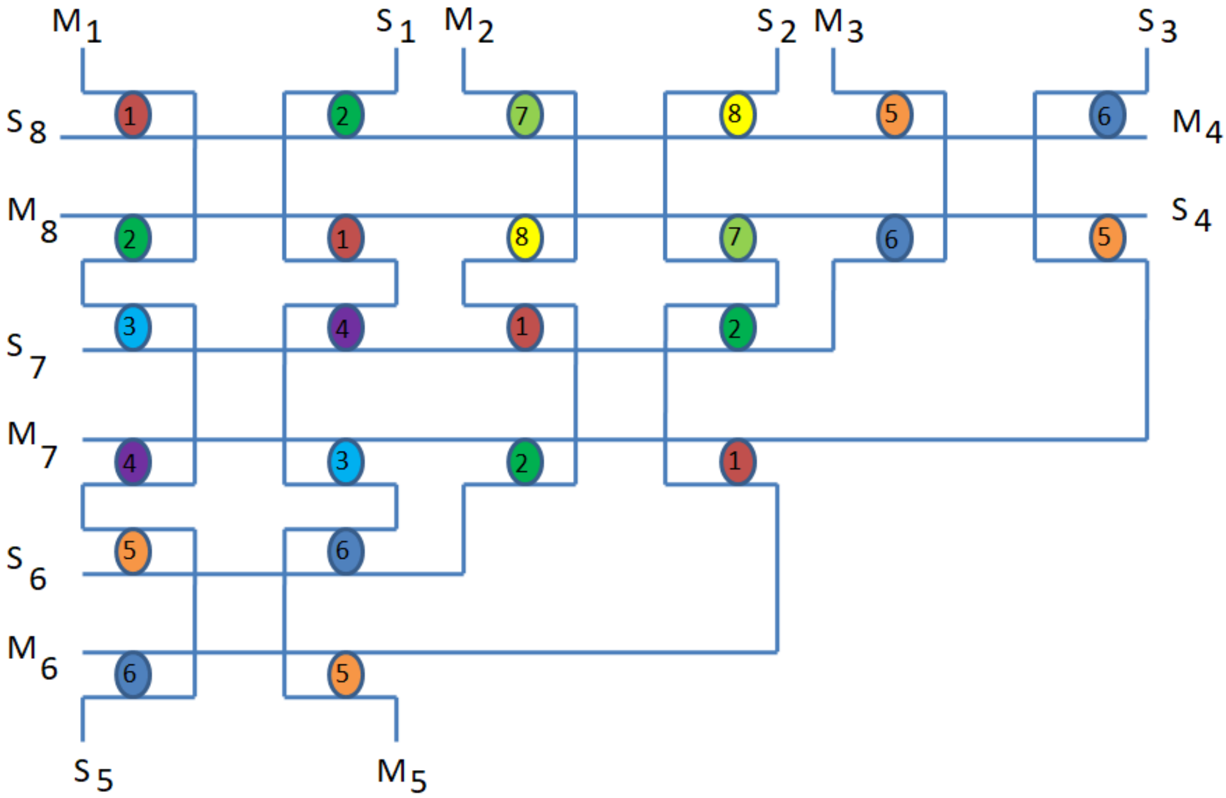

Figure 5.1.: The Light topology for 8x7 Network (Zheng et al., 2021).

r_4, r_6, r_7 and r_8, respectively. Therefore, $v_{worst} = 10$. Thus, the proposed topology achieved a **150%** improvement on the worst case parallelism compared to the Light topology.

In conclusion, the proposed topology performs better than the Light topology in terms of maximizing the worst case parallelism and the aggregate parallelism and minimizing the insertion loss.

Number of path with x Different OFF MRR types	The Light Topology	The Proposed Topology
1 ON MRR and 7 Different OFF MRR types	3	0
1 ON MRR and 6 Different OFF MRR types	3	0
1 ON MRR and 5 Different OFF MRR types	14	8
1 ON MRR and 4 Different OFF MRR types	12	8
1 ON MRR and 3 Different OFF MRR types	6	8
1 ON MRR and 2 Different OFF MRR types	4	8
1 ON MRR and 1 Different OFF MRR type	3	8
1 ON MRR and 0 Different OFF MRRs	3	8
Default paths	8	8

Table 5.3.: Comparison between the Light topology and the proposed topology.

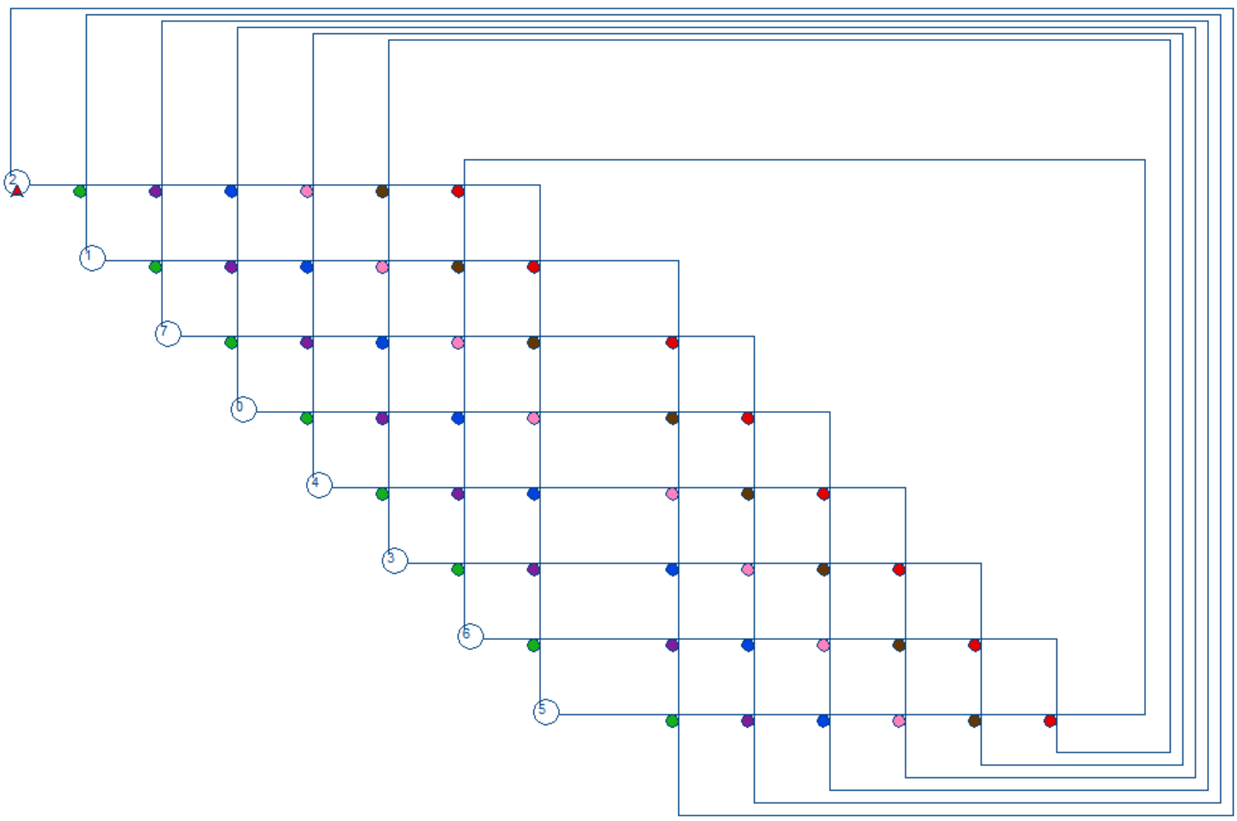

Figure 5.2.: The proposed topology for 8x7 network.

5.4. CustomTopo

CustomTopo is an approach for generating WRONoC topology for application-specific designs. This approach aims to reduce the source usage by minimizing the number of wavelengths and add-drop filters (ADFs). In fact, CustomTopo reduces wavelength and ADF usage by approximately 40% compared to the state-of-the-art method. It additionally aims to reduce power loss. It can be applied not only for symmetric networks but also for asymmetric networks. In this section, CustomTopo is compared to the proposed topology. The communication matrix (5.1) is used for the comparison:

$$C = \begin{array}{c} \\ s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{array} \begin{array}{c} \begin{array}{cccccccc} r_0 & r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \tag{5.1}$$

Based on the communication matrix, CustomTopo generates the matrix 5.2. Each point in the communication matrix with value 0 is transformed into an $\times$. Each point in the communication matrix with value 1 is transformed into an integer value, which represents a wavelength. Default paths between receivers and senders are assigned a value of 0. Then, CustomTopo generates the optimized logic topology as can be seen in Figure 5.3.

$$C = \begin{array}{c} \\ s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{array} \begin{array}{c} \begin{array}{cccccccc} r_0 & r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 \end{array} \\ \begin{pmatrix} \times & 4 & 5 & 6 & 1 & 3 & 2 & 0 \\ 4 & \times & 1 & 5 & 3 & 6 & 0 & 2 \\ 5 & 2 & \times & 1 & 4 & 0 & 6 & 3 \\ 6 & 5 & 2 & \times & 0 & 4 & 3 & 1 \\ 2 & 3 & 4 & 0 & \times & \times & \times & \times \\ 3 & 6 & 0 & 4 & \times & \times & \times & \times \\ 1 & 0 & 6 & 3 & \times & \times & \times & \times \\ 0 & 1 & 3 & 2 & \times & \times & \times & \times \end{pmatrix} \end{array} \tag{5.2}$$

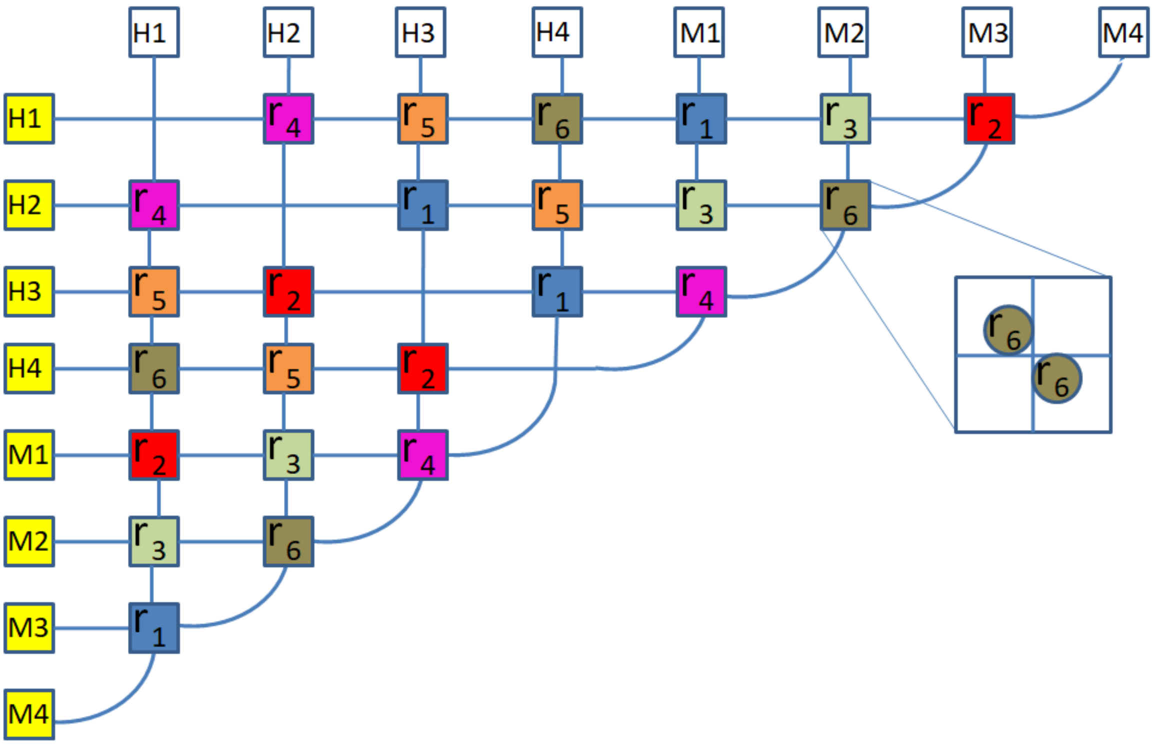

Figure 5.3.: CustomTopo for matrix 5.1.

Figure 5.4 represents the proposed topology when communication matrix 5.1 is the input. As can be seen, the CustomTopo topology uses 48 MRRs while the proposed topology uses only

36 MRRs. This indicates that the proposed topology performs better in terms of minimizing the number of used MRRs.

A drawback of the CustomTopo topology is that the position of each sender is not close to the position of the receiver with the same id, which makes the physical implementation on a chip difficult. Bringing each sender close to its corresponding receiver causes many additional crossing waveguides. Therefore, designers prefer the physical-aware topologies, in which the sender and the receiver of a node are set close to each other, such as in the proposed topology and the Light topology.

The proposed topology is compared with CustomTopo topology in terms of maximizing the parallelism:

- ***The worst case parallelism***: The method presented in (Li et al., 2020) is used for the comparison (alpha=1 and beta=0): For CustomTopo, the aggregate parallelism is v_{total} = 244 and the worst case parallelism is v_{worst} = 6 (Li et al., 2020). For the proposed topology, the radii 12, 23, 20, 16.5, 21.5 and 28 are assigned to r_1, r_2, r_3, r_4, r_5 and r_6, respectively. The aggregate parallelism is v_{total} = 367 and the worst case parallelism is v_{worst} = 10. In conclusion, the proposed topology improves the worst case parallelism by **66.66%** and the aggregate parallelism by **50.4%** compared to Customtopo.

- ***The aggregate parallelism***: Table 5.4 presents a comparison between the CustomTopo topology and the proposed topology. As can be seen, for CustomTopo, $P_{average}$ = 123/40 = 2.79 OFF MRR types/Path. For the proposed topology, $P_{average}$ = 123/40 = 2.07 OFF MRR types/Path. After running the method proposed in (Li et al., 2020) (alpha=0 and beta=1), the worst case parallelism in the proposed topology is 5 and the aggregate parallelism is 452. The worst case parallelism in CustomTopo is 2 and the aggregate parallelism is 288 (Li et al., 2020). Therefore, the proposed topology improves the worst case parallelism by **150%** and the aggregate parallelism by **56.94%** compared to Customtopo, when the target of the method in (Li et al., 2020) is to maximize the aggregate parallelism.

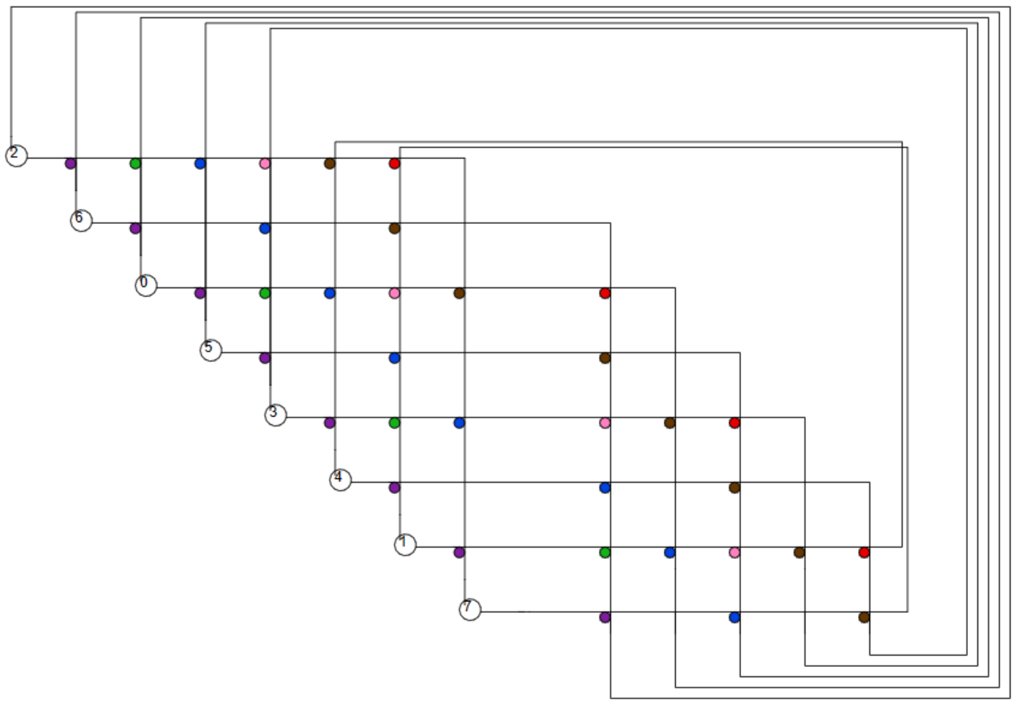

Figure 5.4.: The proposed topology for matrix 5.1.

Number of path with x Different OFF MRR types	CustomTopo	The Proposed Topology
1 ON MRR and 5 Different OFF MRR types	8	4
1 ON MRR and 4 Different OFF MRR types	13	8
1 ON MRR and 3 Different OFF MRR types	7	4
1 ON MRR and 2 Different OFF MRR types	4	8
1 ON MRR and 1 Different OFF MRR type	2	3
1 ON MRR and 0 Different OFF MRRs	2	8
Default paths	8	8

Table 5.4.: Comparison between the Light topology and the proposed topology.

5.5. Snake, λ-router and GWOR Topologies

This section presents the comparison between the proposed topology and Snake, λ-router and GWOR topologies. Different communication matrices and network sizes are used in these comparisons. For each communication matrix, the unused MRRs are removed from the λ-router and GWOR topologies. Table 5.5 presents the comparison results. The method presented in (Li

Matrix	Method	α, β	v_{worst}	v_{total}
$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$	The proposed topology	1, 0	18	148
		0, 1	18	148
	GWOR	1, 0	16	139
		0, 1	14	148
$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$	The proposed topology	1, 0	15	180
		0, 1	14	204
	Snake	1, 0	11	134
		0, 1	10	172
	$\lambda - router$	1, 0	10	120
		0, 1	10	160
$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$	The proposed topology	1, 0	13	266
		0, 1	9	314
	GWOR	1, 0	8*	240*
		0, 1	8*	253*
	$\lambda - router$	1, 0	5*	171*
		0, 1	4*	180*
Matrix 5.1	The proposed topology	1, 0	10	367
		0, 1	5	452
	GWOR	1, 0	5*	292*
		0, 1	3*	335*
	$\lambda - router$	1, 0	2*	213*
		0, 1	0*	279*

Table 5.5.: Comparison results.

et al., 2020) is used to compare the performance of the topologies. For the last communication matrix, the time limit was set to 8 hours. For all other experiments, the time limit was set to 6 hours. If the method presented in (Li et al., 2020) could not terminate within the given time, the provided results are intermediate results (These results are marked with *). As can be seen, the proposed topology outperforms the other topologies in all cases in terms of maximizing the worst-case parallelism and the aggregate parallelism.

5.6. The State-of-the-art Parallelism-Aware Topology

The state-of-the-art parallelism-aware topology (Chen et al., 2023) is compared with the proposed topology in terms of reducing the number of MRR types, improving the aggregate parallelism and the worst case parallelism. Since both topologies satisfy Property 3.1, the path with the highest number of OFF MRR types represents the worst case parallelism. Equation 5.2 is used to determine the number of OFF MRR types in the path that contains the highest number of OFF MRRs, where P is the set of all paths that contain one ON MRR and $N_{OFF\_MRR\_in\_p}$ represents the number of OFF MRR types in the path p. To compare the worst case parallelism between two topologies T_1 and T_2, the values of $P_{worst\_T_2}$ and $P_{worst\_T_1}$ should be compared. The topology with lower P_{worst} performs better in terms of maximizing the worst case parallelism.

$$P_{worst} = \max_{p \in P} N_{OFF\_MRR\_in\_p} \tag{5.2}$$

In addition, to determine which of the two topologies T_1 and T_2 is better in terms of maximizing the aggregate parallelism, the values of $P_{average\_T_2}$ and $P_{average\_T_1}$ should be compared. The topology with the lower $P_{average}$ performs better.

For a fair comparison between the two topologies, the approach presented in (Chen et al., 2023) is re-implemented. For the comparison between the two topologies, the variation of the laser's operating wavelength x is considered. This variation is created by the variation in driving current and temperature (Chen et al., 2023). As in the experiments results presented in (Chen et al., 2023), a wavelengths x conflicts with another wavelength y when the absolute difference between the two wavelengths (x and y) is smaller than δ + x. In these comparisons, x= 0.5nm (Peano et al., 2016).

The communication matrix 5.1 is used for the first comparison. The communication matrices C_1 and C_2 presented in 5.3 are used for the second and third comparisons. The method in (Li et al., 2020) is used for the comparisons. In these comparisons, the aim of the method in (Li et al., 2020) is to maximize the worst case parallelism (alpha is set to 1 and beta is set to 0). In these comparisons, the parallelism of a default path is 20.

$$C_1 = \begin{array}{c} \\ s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} r_0 & r_1 & r_2 & r_3 & r_4 & r_5 & r_6 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array} \quad C_2 = \begin{array}{c} \\ s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{array} \begin{array}{c} \begin{array}{cccccc} r_0 & r_1 & r_2 & r_3 & r_4 & r_5 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array} \tag{5.3}$$

Table 5.6 presents the comparison results between the proposed topology and the topology in (Chen et al., 2023). As can be seen, for the first communication matrix, the two topologies perform similarly in terms of maximizing the worst case parallelism. However, the proposed topology improves the aggregate case parallelism by **11.959%** compared to the topology in (Chen et al., 2023). For the second communication matrix, the proposed topology improves the worst case parallelism by **25%** and the aggregate parallelism by **23.337%** compared to the topology in (Chen et al., 2023). For the third communication matrix, the proposed topology improves the worst case parallelism by **41.66%** and the aggregate parallelism by **23.29%** compared to the topology in (Chen et al., 2023). In all these experiments, the target of the approach in (Li et al., 2020) was to maximize the worst case parallelism. In conclusion, although the topology in (Chen et al., 2023) is designed mainly to maximize the worst case parallelism, there are several cases, where the proposed topology performs better than the topology in (Chen et al., 2023) in terms of maximizing the worst case parallelism. Thus, the proposed topology outperforms the topology in (Chen et al., 2023) in terms of maximizing the worst case parallelism. In all these cases, the proposed topology maximizes the worst case parallelism (The results are optimal in terms of maximizing the worst case parallelism because the number of MRR types is minimized).

The proposed topology reduced the number of used MRRs and the number of MRR types used in the topology. In the second and the third comparisons, the proposed topology reduced the number of used MRR types by 1 compared to the state-of-the-art parallelism-aware topology (Chen et al., 2023). However, the state-of-art topology reduces the insertion loss by 0.08 dB compared to the proposed topology. For the first communication matrix, the number of MRR types used in the two topologies is 5. In addition, the maximum insertion loss is 0.9 dB in the tow topologies. The average insertion loss in the proposed topology is 0.649 dB, while the average insertion loss in the state-of-the-art topology is 0.6831 dB. Thus, for the first communication matrix, the proposed topology performs better in terms of minimizing the average insertion loss.

Both methods terminate in less than a minute in all cases. For example, for the second communication matrix, the proposed topology terminates in 19.79 s, while the state-of-the-art topology terminates in 11,94 s (Note that the proposed topology aims to maximize the worst case parallelism and the aggregate parallelism at the same time unlike the state-of-the-art topology).

The ILP formulation of the topology in (Chen et al., 2023) aims only to reduce the upper bound of the number of different OFF MRR types in all paths. It does not aim to maximize the number of default paths or the number of paths with one ON MRR and zero OFF MRRs. Therefore, the proposed topology performs better in terms of maximizing the

Matrix	Topology	Figure	Max IL	$P_{average}$	P_{worst}	v_{worst}	v_{total}
Matrix 5.1	The proposed topology	5.4	0.9 dB	1.77	5	6	440
	in (Chen et al., 2023)	5.9	0.9 dB	2.31	5	6	393
C_1 *in* 5.3	The proposed topology	5.5	0.9 dB	1.147	3	10	417
	in (Chen et al., 2023)	5.6	0.82 dB	1.705	4	8	338
C_2 *in* 5.3	The proposed topology	5.8	0.82 dB	0.238	1	17	307
	in (Chen et al., 2023)	5.7	0.74 dB	0.588	2	12	249

Table 5.6.: The comparison results between the proposed topology and the topology in (Chen et al., 2023)

aggregate parallelism.

In conclusion, the proposed topology outperforms the topology in (Chen et al., 2023) in terms of improving the worst case parallelism and the aggregate parallelism.

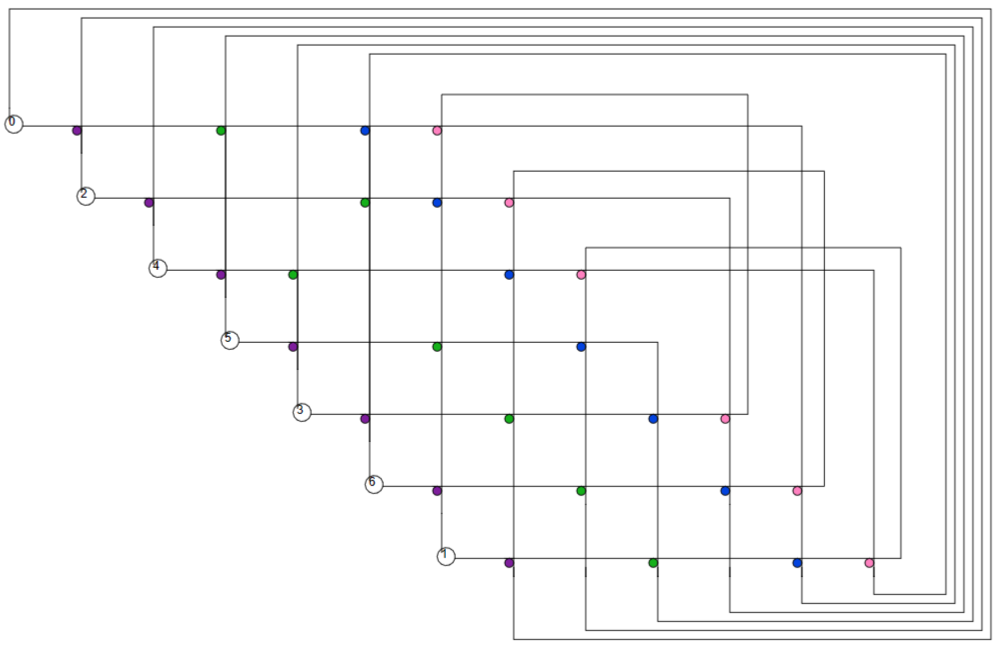

Figure 5.5.: The proposed topology for the first matrix in 5.3.

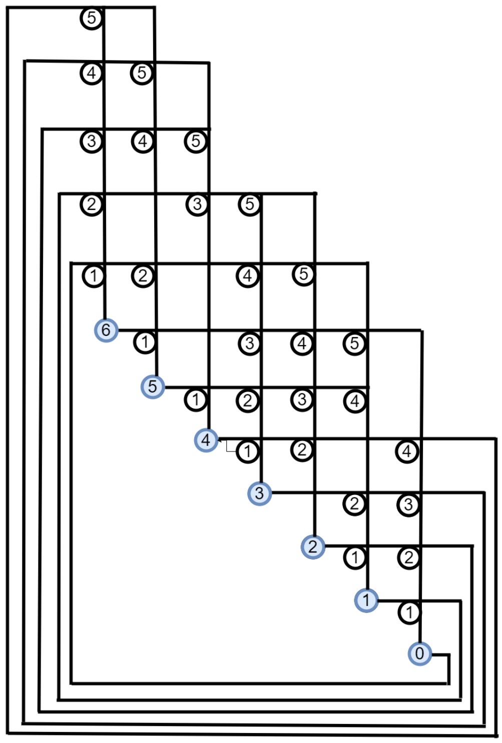

Figure 5.6.: The topology in (Chen et al., 2023) for the first matrix in 5.3.

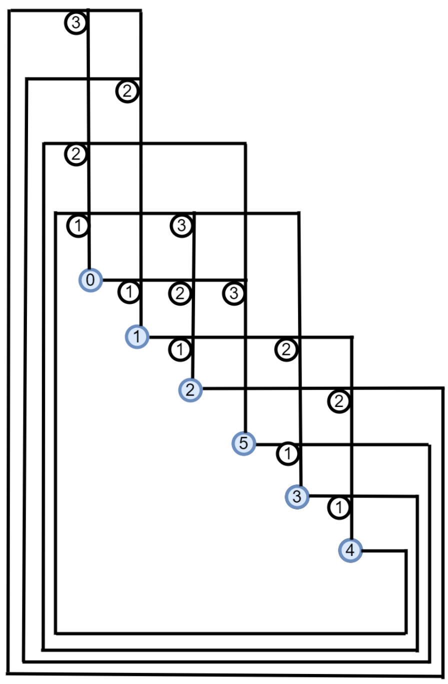

Figure 5.7.: The topology in (Chen et al., 2023) for the second matrix in 5.3.

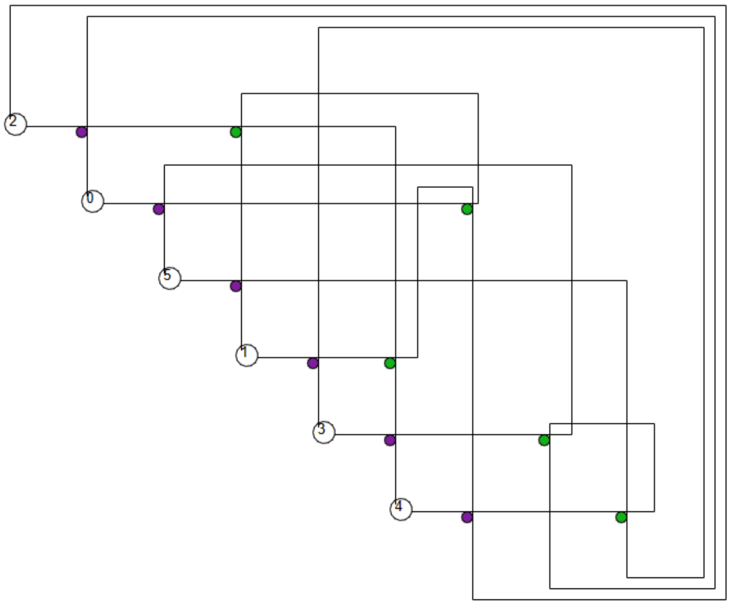

Figure 5.8.: The proposed topology for the second matrix in 5.3.

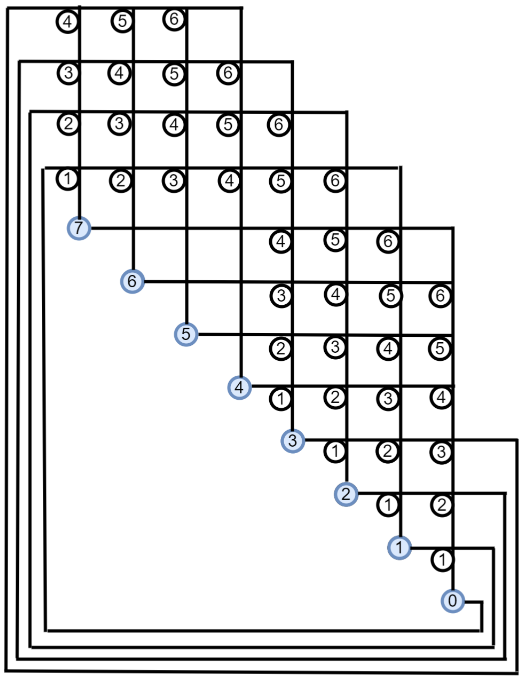

Figure 5.9.: The topology in (Chen et al., 2023) for matrix 5.1.

5.7. SNR

In this section, the SNR of the proposed topology is calculated. In addition, the SNR of the proposed topology is compared with the SNR of the other topologies. As mentioned in Section 2.4, the SNR can be calculated using the Equation 5.3 (Zheng et al., 2021). For simplification, the calculation takes into account only the first-order noise introduced by the optical signals. The noise signals of value less than -41 dB P_{in} are ignored.

$$SNR(x) = 10log\frac{P_{output}^{\lambda_n}(x)}{P_{noise}^{\lambda_n}(x)} \tag{5.3}$$

Figure 5.10 represents the proposed topology for a 4x4 full connectivity matrix. First, the output power of the signal (1, 2) at the receiver 2 is calculated. The output power signal (1, 2) is represented by $P_{output}^{\lambda_{green}}(1,2)$. The optical signal (1, 2) is transmitted through one ON CSE ($m_{1,2}$) and two OFF CSEs. Therefore, $P_{output}^{\lambda_{green}}(1,2) = L_{cse,off}^2 L_{cse,on} P_{in}$ = -0.58 dB P_{in}. The power of all crosstalk signals that reach the receiver 2 and have the wavelength λ_{green} is represented by $P_{noise}^{\lambda_{green}}(1,2)$ (Zheng et al., 2021).

The dashed lines represent the crosstalk signals reaching core 2 and having wavelength λ_{green}. As can be seen, there are two crosstalk signals N1 and N2. The crosstalk signal N1 is introduced when optical signals are transmitted from core 3 to core 0. The crosstalk signal N1 (the dashed line) is transmitted along the default path and reaches the core id 2, as can be seen in Figure 5.10. The CSEs in the topology support only 90-degree rotation. Therefore, the crosstalk signal does not change direction by 270-degree when it enters the second green MRR ($m_{1,2}$) from port A (from above) (see Figure 5.10). When the optical signal enters the CSE via Port A, the CSE behaves like a waveguide crossing and the optical signal continues to transmit along the original waveguide without changing the direction (independent of the wavelength of the signal). Therefore, P_{N1} can be calculated as follows: $P_{N1} = C_{p,on} L_{cse,off}^6 P_{in}$.

The crosstalk signal N2 is introduced when optical signals are transmitted from core 2 to core 3. The crosstalk signal N2 (the dashed line) is generated when the optical signals ignores the violet MRR (see Figure 5.10). P_{N2} can be calculated as follows: $P_{N2} = (C_{p,off} + L_{p,off}^2 C_{cr}) P_{in}$

$P_{noise}^{\lambda_{green}}(1,2) = P_{N2} + P_{N2} = C_{p,on} L_{cse,off}^6 P_{in} + (C_{p,off} + L_{p,off}^2 C_{cr}) P_{in}$. Therefore, $SNR^{\lambda_{green}}(1,2) = 10log\frac{P_{output}^{\lambda_n}(x)}{P_{noise}^{\lambda_n}(x)} = 21.22$.

In summary, the SNR of the signals (0, 0), (1, 1), (2, 2) and (3, 3) is 24.605. The SNR of the signals (0, 1), (1, 2), (2, 3) and (3, 0) is 21.22. The SNR of the signals (0, 2), (1, 3), (2, 0) and (3, 1) is 19.37. The SNR of the default paths is 19.8.

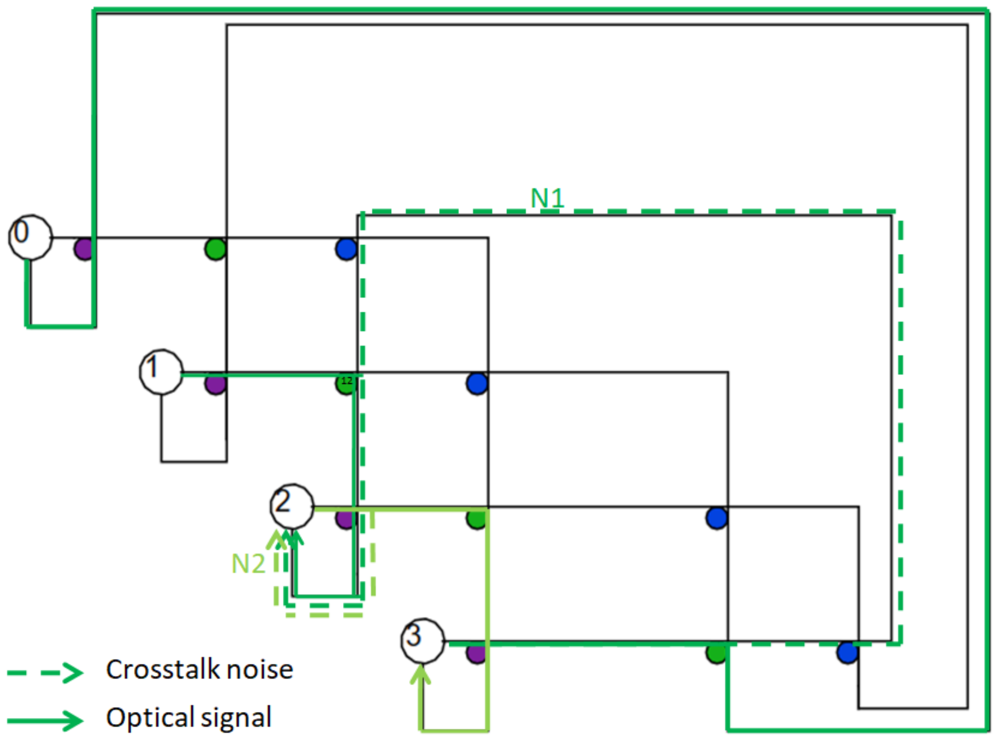

Figure 5.10.: An example of the crosstalk noise in the proposed topology

Table 5.7 represents the SNR in different routers for different network sizes (Zheng et al., 2021). As can be seen, the average SNR and the worst case SNR of the proposed topology is always higher than these values of the other topologies. In conclusion, the proposed topology outperforms λ-router and Light in terms of improving the average SNR and the worst case SNR.

Size	SNR	$\lambda-$router	Light	The proposed topology
4*3	The average SNR/dB	20.11	22.11	22.49
	The worst case SNR/dB	17.14	19.9	21.41
8*7	The average SNR/dB	14.672	18.875	19.16
	The worst case SNR/dB	13.443	14.218	16.41
16*15	The average SNR/dB	10.749	14.48	16.4
	The worst case SNR/dB	10.12	10.1	12.38

Table 5.7.: SNR in Light, $\lambda-$router and the proposed topology.

6. Conclusions

This work considers the communication parallelism in the topology design stage. More specifically, the proposed approach aims to design a topology that maximizes the worst case parallelism as well as the aggregate parallelism for a given communication matrix. The proposed topology improves the parallelism for full connectivity and for customized connectivity. In addition, this work aims to reduce the insertion loss. The proposed topology is the first to present an approach that aims to maximize the aggregate parallelism during topology generation.

To achieve these results, an algorithm was implemented. The algorithm consists of three main steps. In the first step, a network template is proposed. The design of the network template matches the physical layout constraints. Therefore, the proposed topology can avoid unnecessary waveguide crossings during physical implementation, resulting in a reduction in insertion loss. In the second step, an integer linear programming model is proposed to determine the position of the cores and the MRRs in the template network, to select the default paths and to assign a type to each MRR in the topology. In the third step, the algorithm uses the results of the previous step to design and draw the topology automatically and save it in an image format. In this step, the number of waveguide crossings is reduced in order to minimize the insertion loss.

Finally, in order to evaluate the performance of the proposed topology, multiple tests were performed to compare the proposed topology with other topologies such as CustomTopo, Light, Snake and the state-of-the-art parallelism-aware topology (Chen et al., 2023) in terms of improving the parallelism and reducing the insertion loss. Several random communication matrices are used throughout these comparisons. Experimental results show that the proposed topology outperforms the state-of-the-art parallelism-aware topology (Chen et al., 2023) in terms of maximizing the worst case parallelism. The selection of default paths in the proposed topology is very flexible and does not depend on the position of the cores unlike the selection of default paths in the state-of-the-art work (Chen et al., 2023). The selection of the default paths in the proposed topology aims to reduce the number of MRR types, leading to improving the worst case parallelism. Furthermore, the proposed topology performs better than the state-of-the-art topologies in terms of improving the aggregate parallelism. In addition, the topology is compared with the Light typology in terms of reducing the power loss, improving the SNR and the parallelism. Although the Light topology is implemented to reduce the insertion loss, the proposed topology performs better not only in terms of improving the parallelism but also in terms of reducing the insertion loss and improving the SNR.

References

Bogaerts, W., De Heyn, P., Van Vaerenbergh, T., De Vos, K., Kumar Selvaraja, S., Claes, T., ... Baets, R. (2012). Silicon microring resonators. *Laser & Photonics Reviews*, *6*(1), 47–73.

Briere, M., Girodias, B., Bouchebaba, Y., Nicolescu, G., Mieyeville, F., Gaffiot, F., & O'Connor, I. (2007). System level assessment of an optical noc in an mpsoc platform. In *2007 design, automation & test in europe conference & exhibition* (pp. 1–6).

Chan, J., Hendry, G., Bergman, K., & Carloni, L. P. (2011). Physical-layer modeling and system-level design of chip-scale photonic interconnection networks. *IEEE Transactions on computer-aided design of integrated circuits and systems*, *30*(10), 1507–1520.

Chen, K.-C., Chen, Y.-L., Lu, Y.-S., & Chang, Y.-W. (2023). Toward parallelism-optimal topology generation for wavelength-routed optical noc designs. In *2023 60th acm/ieee design automation conference (dac)* (pp. 1–6).

Ding, W., Tang, D., Liu, Y., Chen, L., & Sun, X. (2010). Compact and low crosstalk waveguide crossing using impedance matched metamaterial. *Applied Physics Letters*, *96*(11).

Fitzpatrick, M. (2023). *Pyqt5 vs pyside2.* https://www.pythonguis.com/faq/pyqt5-vs-pyside2/.

GUROBI, O. (2020). *Gurobi optimizer quick start guide.* Version.

Koohi, S., & Hessabi, S. (2012). Scalable architecture for a contention-free optical network on-chip. *Journal of Parallel and Distributed Computing*, *72*(11), 1493–1506.

Le Beux, S., Li, H., Nicolescu, G., Trajkovic, J., & O'Connor, I. (2014). Optical crossbars on chip, a comparative study based on worst-case losses. *Concurrency and Computation: Practice and Experience*, *26*(15), 2492–2503.

Le Beux, S., Trajkovic, J., O'Connor, I., Nicolescu, G., Bois, G., & Paulin, P. (2011). Optical ring network-on-chip (ornoc): Architecture and design methodology. In *2011 design, automation & test in europe* (pp. 1–6).

Li, M., Tseng, T.-M., Bertozzi, D., Tala, M., & Schlichtmann, U. (2018). Customtopo: A topology generation method for application-specific wavelength-routed optical nocs. In *2018 ieee/acm international conference on computer-aided design (iccad)* (pp. 1–8).

Li, M., Tseng, T.-M., Tala, M., & Schlichtmann, U. (2020). Maximizing the communication parallelism for wavelength-routed optical networks-on-chips. In *2020 25th asia and south pacific design automation conference (asp-dac)* (pp. 109–114).

Lin, B.-C., & Lea, C.-T. (2012). Crosstalk analysis for microring based optical interconnection networks. *Journal of lightwave technology*, *30*(15), 2415–2420.

Manolatou, C., & Haus, H. A. (2002). *Passive components for dense optical integration.* Springer Science & Business Media.

Parini, A., Ramini, L., Bellanca, G., & Bertozzi, D. (2011). Abstract modelling of switching elements for optical networks-on-chip with technology platform awareness. In *Proceedings of the fifth international workshop on interconnection network architecture: On-chip, multi-chip* (pp. 31–34).

Peano, A., Ramini, L., Gavanelli, M., Nonato, M., & Bertozzi, D. (2016). Design technology for fault-free and maximally-parallel wavelength-routed optical networks-on-chip. In *2016 ieee/acm international conference on computer-aided design (iccad)* (pp. 1–8).

Preston, K., Sherwood-Droz, N., Levy, J. S., & Lipson, M. (2011). Performance guidelines for wdm interconnects based on silicon microring resonators. In *Cleo: 2011-laser science to photonic applications* (pp. 1–2).

Pyqt5. (2023). Retrieved 2023-10-14, from `https://pypi.org/project/PyQt5/`

Ramini, L., Bertozzi, D., & Carloni, L. P. (2012). Engineering a bandwidth-scalable optical layer for a 3d multi-core processor with awareness of layout constraints. In *2012 ieee/acm sixth international symposium on networks-on-chip* (pp. 185–192).

Ramini, L., Grani, P., Bartolini, S., & Bertozzi, D. (2013a). Contrasting wavelength-routed optical noc topologies for power-efficient 3d-stacked multicore processors using physical-layer analysis. In *2013 design, automation & test in europe conference & exhibition (date)* (pp. 1589–1594).

Ramini, L., Grani, P., Bartolini, S., & Bertozzi, D. (2013b). Contrasting wavelength-routed optical noc topologies for power-efficient 3d-stacked multicore processors using physical-layer analysis. In *2013 design, automation & test in europe conference & exhibition (date)* (pp. 1589–1594).

Savage turtle. (2023). Retrieved 2023-09-30, from `https://pypi.org/project/svg-turtle/`

Tala, M., & Bertozzi, D. (2018). Understanding the design space of wavelength-routed optical noc topologies for power-performance optimization. In *2018 ifip/ieee international conference on very large scale integration (vlsi-soc)* (pp. 255–260).

Tan, X., Yang, M., Zhang, L., Jiang, Y., & Yang, J. (2011). On a scalable, non-blocking optical router for photonic networks-on-chip designs. In *2011 symposium on photonics and optoelectronics (sopo)* (pp. 1–4).

Truppel, A., Tseng, T.-M., Bertozzi, D., Alves, J. C., & Schlichtmann, U. (2019). Psion: Combining logical topology and physical layout optimization for wavelength-routed onocs. In *Proceedings of the 2019 international symposium on physical design* (pp. 49–56).

Vantrease, D., Schreiber, R., Monchiero, M., McLaren, M., Jouppi, N. P., Fiorentino, M., …

Ahn, J. H. (2008). Corona: System implications of emerging nanophotonic technology. *ACM SIGARCH Computer Architecture News*, *36*(3), 153–164.

Wang, Z., Pang, Z., Yang, P., Xu, J., Chen, X., Maeda, R. K., ... Wang, Z. (2017). Moca: An inter/intra-chip optical network for memory. In *Proceedings of the 54th annual design automation conference 2017* (pp. 1–6).

Werner, S., Navaridas, J., & Luján, M. (2017). A survey on optical network-on-chip architectures. *ACM Computing Surveys (CSUR)*, *50*(6), 1–37.

Xie, Y., Nikdast, M., Xu, J., Wu, X., Zhang, W., Ye, Y., ... Liu, W. (2012). Formal worst-case analysis of crosstalk noise in mesh-based optical networks-on-chip. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *21*(10), 1823–1836.

Yan, H., Xie, Y., Yang, X., & Song, T. (2020). A novel algorithm for reducing the power loss of routing paths in onocs. In *2020 international conference on wireless communications and signal processing (wcsp)* (pp. 325–330).

Zhang, L., Man, Y., Tan, X., Yang, M., Hu, T., Yang, J., & Jiang, Y. (2014). On reducing insertion loss in wavelength-routed optical network-on-chip architecture. *Journal of Optical Communications and Networking*, *6*(10), 879–889.

Zheng, Z., Li, M., Tseng, T.-M., & Schlichtmann, U. (2021). Light: A scalable and efficient wavelength-routed optical networks-on-chip topology. In *Proceedings of the 26th asia and south pacific design automation conference* (pp. 568–573).