Research Article Vol. 12, No. 9 / September 2024 / Photonics Research 1999

Routing impact of architecture and damage in programmable photonic meshes

Ferre Vanden Kerchove,[1,*] Didier Colle,[1] Wouter Tavernier,[1] Wim Bogaerts,[2,3] and Mario Pickavet[1]

[1]IDLab, Department of Information Technology, Ghent University - imec, 9052 Ghent, Belgium
[2]Photonics Research Group, Department of Information Technology, Ghent University - imec, 9052 Ghent, Belgium
[3]Center of Nano- and Biophotonics, Ghent University, 9052 Ghent, Belgium
*Corresponding author: ferre.vandenkerchove@ugent.be

Received 26 March 2024; revised 7 June 2024; accepted 3 July 2024; posted 5 July 2024 (Doc. ID 523986); published 29 August 2024

Programmable photonic integrated circuits (PPICs) emerge as a novel technology with an enormous potential for ground-breaking innovation. Different architectures are currently being considered that dictate how waveguides should be connected to realize a broadly usable circuit. We focus on the effect of varying connectivity architectures on the routing of light. Three types of uniform meshes are studied, and we introduce a newly developed mesh that is called ring-connected straight lines. We provide an analytical formula to calculate exact distances in these meshes and introduce several metrics relating to routing to compare these meshes. We show that hexagonal tiles are the most promising, but the ring-connected straight lines architecture has a use case as well. Besides this, the effect of defect couplers is also studied. We find that the effects of these failures vary greatly by type and severity on the routability of the mesh.

https://doi.org/10.1364/PRJ.523986

1. INTRODUCTION

Photonic integrated circuits (PICs) are implemented on the surface of a chip, where they guide and manipulate light through waveguides, beam couplers, modulators, lasers, and photodetectors. Presently, most PICs are custom-designed application-specific photonic integrated circuits (ASPICs), which come with a long development cycle and high fabrication costs. There is a growing interest, akin to the evolution in electronics, in programmable PICs as the next logical progression. This will lower the threshold for testing new photonic functionality [1,2]. These chips can be programmed to execute various functions, eliminating the need for custom designs. The programmability is facilitated through tunable couplers connecting two waveguides, each with different modes governing light flow. Bar mode lets the light continue in its current waveguide, whereas in cross mode the light crosses over to the neighboring waveguide. An intermediate coupling mode distributes the light over both waveguides. By smartly choosing the correct mode for each coupler, a wide variety of functions are realizable. Programmable PICs could give a tremendous speed-up in prototyping, enabling new potential breakthroughs in areas such as quantum optics [3,4], communication [5,6], and machine learning [7,8]. In Ref. [9], the author argues that the potential of photonics can only be realized through an efficient combination of several photonic advantages. Programmable photonics has the potential to substantially improve the feasibility of achieving this combination. A showcase of the potential for programmable photonics is given in Ref. [10].

In Section 2, we review related work. Section 3 introduces the necessary definitions. In Section 4, we present different architectures, detailing the layout and connectivity of waveguides through couplers. We analyze how efficiently different architectures can route single signals in Section 5.A and which restriction they impose. Section 5.B provides an analytical distance for each tile shape. To conclude this part, in Section 5.C, we evaluate whether certain layouts are more suitable for accommodating multiple paths concurrently. Additionally, given that a percentage of components fail, we examine the impact of these failures on routability. Section 6 discusses this impact and provides a rule of thumb to determine the necessary redundancy.

2. RELATED WORK

In this paper, we discuss two topics. First, we compare the routability of different architectures for meshes. This is in contrast to Refs. [11,12] where the authors study the effect of different architectures on optical functionality. In Ref. [13], the authors take an analytical approach and investigate which path lengths are realizable in meshes with either square or hexagonal tile shapes. Complementary to their work, we provide analytical formulas for the distance between two points in the different mesh architectures. Although hexagonal tiles are the most

popular, this is definitely not a settled debate. For example, Refs. [13–15] use square tiles as their main tile shape.

The other is, to the best of our knowledge, not yet studied before and concerns the effect of component failure on routing. Several papers investigate circuit yields [16] and the impact of imperfect coupling [17–19], but the effects of coupler failure on routing remain largely undiscussed. Using the integer program and routing algorithm described in Ref. [20], we assess the impact of failure on the routing capabilities of photonic meshes.

3. DEFINITIONS AND KEY METRICS

A photonic mesh comprises waveguides and tunable couplers that manipulate the flow of light between waveguides. Each coupler can assume a bar, cross, or coupling state, allowing the possible connections to be represented as a graph, as illustrated in Fig. 1 [21]. The external ports, depicted as colored nodes, serve as interfaces for coupling light into and out of the mesh. The couplers of one hexagon are indicated as well. The mesh directs light from the source port to the destination port through the use of couplers and waveguides. A commodity is defined as a pair of ports, forming a source-destination pair. The objective, given a list of commodities, is to identify an appropriate route within the mesh for each commodity, ensuring a path from source to destination, as depicted in Fig. 3 (shown later). A routing is valid when no two paths utilize the same routing resources. The translation of each mesh into a directed graph facilitates the application of various graph-based routing algorithms. For more details on this translation, see Ref. [21]. A more comprehensive discussion of this problem, including a rigorous problem statement and various solutions, is provided by Ref. [20].

In this paper, we investigate the effect on routing that different architectures have. We limit ourselves to recirculating meshes, as these provide greater flexibility in comparison to forward-only meshes [2]. We measure through key metrics the ability of a mesh to route. These are (1) the number of signals, i.e., commodities, that a mesh can route at the same time. The greater number of commodities shows that the mesh is more flexible, as it can be used more intensely. (2) The average path length, expressed in the number of couplers that are present in the path. We argue that it is logical to measure the path length in the number of couplers that are present in the path, in contrast to measuring the absolute length of a path, which ranges from hundreds of micrometers up to a few millimeters. As discussed in Ref. [22], the main figures of merit for paths in photonic circuits are insertion loss or attenuation, power consumption, basic unit length, which is the sum of the tunable coupler length and the arc length of the access waveguides, and the basic unit delay [23]. Given the total size of the mesh, as well as the speed ($\approx 0.5c$) at which signals propagate [24], the total length does not contribute a lot to latency, nor is it a large source of attenuation. Using figures from Ref. [10], the propagation losses are around 2 dB/cm, which is a largely negligible source of attenuation. The insertion loss, however, is 0.48 dB per coupler. This is highly relevant, as even in smaller meshes the architecture can cause a difference in path length of six couplers, which equates to an additional 3 dB attenuation, or around 50% reduction in signal strength. In larger meshes this can easily increase to 12 or more couplers, giving a 75% reduction in optical power. Power consumption is an important measure besides attenuation. The more couplers in a path, the higher the energy consumption is to drive all the couplers in this path. How much influence this has, relies on the actual implementation of the couplers. Couplers are also a source of imperfection and production variations. Fewer couplers in a path mean less cumulative production variation. These reasons explain why all lengths are expressed in the number of couplers in the path, as opposed to micrometers. We now introduce the various architectures that are studied in this paper.

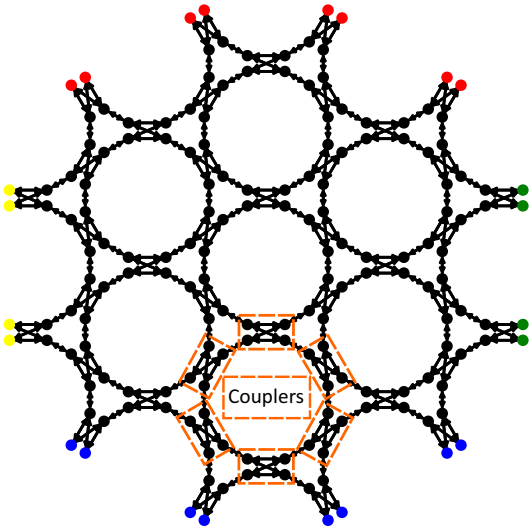

Fig. 1. Graph representation of the possible connections in a seven-tile hexagonal waveguide mesh. The overall contour of the mesh is hexagonal, and the tiles are hexagons as well. There are 42 couplers. The hexagonal shape of the tiles is not completely straightforward but becomes apparent when one focuses on the six couplers making up the tile.

4. DIFFERENT SHAPES

A distinction is made between tile shapes and the overall contour of the mesh. As expected, the tile shape refers to the shape of a single tile. These are triangles, squares, hexagons, or ring-connected straight lines in this paper. In contrast, the overall contour of the mesh refers to how the different tiles are laid out, and which pattern they form. Often this is square, but it is rather natural to arrange hexagonal tiles in a hexagonal contour as well. For example, in Fig. 1, the overall contour is hexagonal, and the tiles are hexagons as well. Compare this to Fig. 2(c), where the overall contour is rectangular while the tiles are still hexagons.

As mentioned before, a port is the connection of a mesh with the outside. An arm of a coupler not connected to further waveguides is considered a port. In the various figures, these ports are indicated by colored nodes. In this paper, the overall contour is always assumed to be square, and we study the impact of different tile shapes. This contour was chosen because all tile shapes lend themselves well to form a square contour. Furthermore, the three most common choices are square, rectangular, and hexagonal contours. We argue that the results in

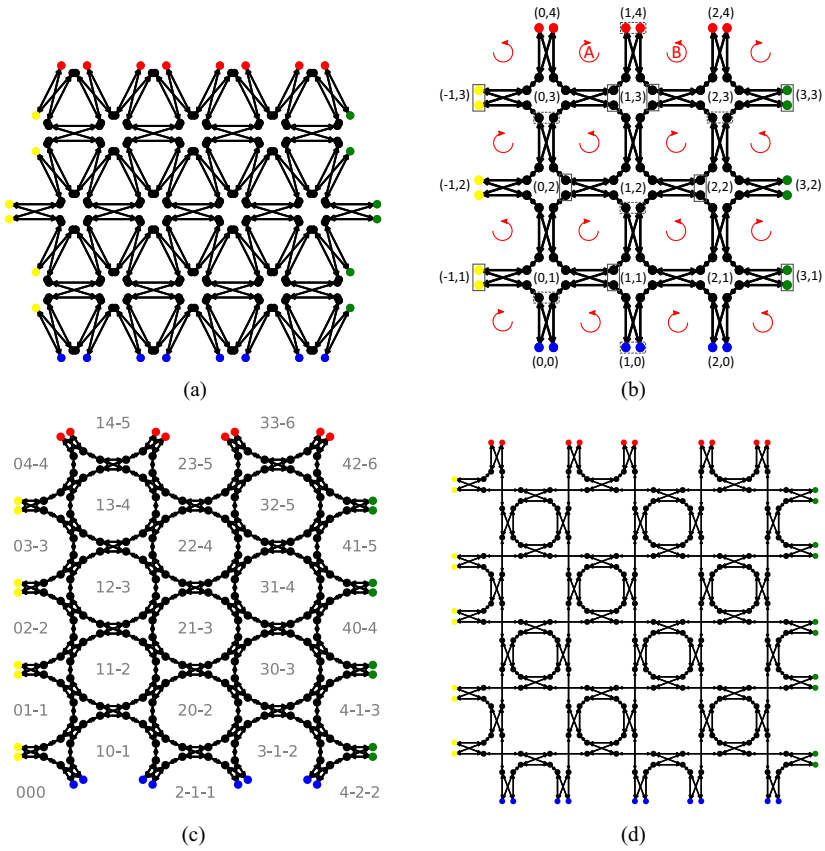

Fig. 2. Four studied architectures and the number of couplers that compose them organized in a square contour. (a) A mesh with triangular tiles: 45 couplers. (b) A mesh with square tiles: 24 couplers. (c) A mesh with hexagonal tiles: 56 couplers. (d) A mesh with ring-connected straight lines: 60 couplers.

this paper are the same when the contour is rectangular instead. This is based on the fact that square and rectangle are rather similar. Except for the hexagonal tile shape, the other tiles do not lend themselves well to form a hexagonal contour, which would reduce the validity of the comparison.

The first three tile shapes are straightforward, consisting of the three regular tessellations: triangles, squares, and hexagons, as depicted in Fig. 2(a), Fig. 2(b), and Fig. 2(c), respectively. Whenever two tiles share a side, a coupler is placed there. We propose a fourth novel alternative, ring-connected straight lines (RCSLs), illustrated in Fig. 2(d), which offers several benefits.

When comparing paths from point A to point B using different tile shapes, the directness of the paths varies. For instance, in a mesh with triangular tiles, some paths are inefficient, as shown in Fig. 3. After each coupler, a path bends by 120°, causing it to zigzag through the mesh. Particularly, in the left path of Fig. 3, for every four couplers, only one proceeds straight to the destination, while the other three accommodate these 120° bends. Similarly, in a mesh with square tiles, if the source and destination are vertically aligned, every other coupler moves left or right, unnecessarily lengthening the path.

This phenomenon is further examined with the following consideration: looking at a single commodity in a mesh, the light's flow is significantly deviated after each coupler due to the natural bends in the mesh-on average, 60° with hexagonal tiles, 90° with square tiling, and 120° with triangles. This deviation results in inefficient routing, as a straight path is never achievable, thus consuming more routing resources and leaving fewer resources for other paths. Consequently, the total number of routable light signals in the mesh is reduced.

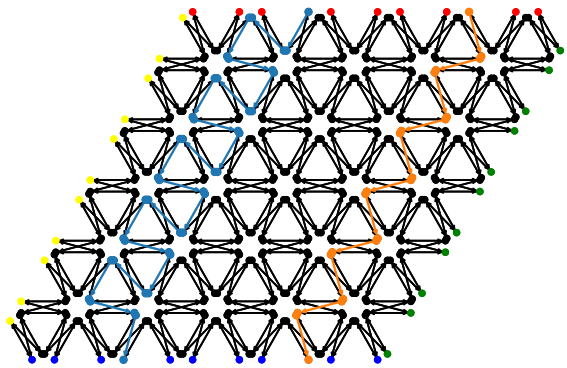

Fig. 3. Mesh with a parallelogram contour with triangular tiles and two commodities with a shortest path. The leftmost path seems to be rather wasteful, needing three additional couplers for every coupler on the straight line from source to destination.

Based on this insight, we explore whether an alternative tiling could improve performance. Ensuring that each coupler allows for either 0° deviation (going straight) or 90° deviation (turning left or right) achieves this. Each coupler is positioned to connect one side to a straight waveguide and the other side to a waveguide making a 90° turn. The key question is how to decide the direction of the bend for each coupler. A balanced distribution of left and right bends is necessary to maximize general routability.

After evaluating different alternatives, the most promising scheme emerged, as shown in Fig. 2(d). Along a straight line, an alternating pattern of left and right bends is chosen. If the desired bend is unavailable at a coupler, the next coupler along the straight waveguide bends in the opposite direction. Adjacent straight lines start with bends in opposite directions, forming a square pattern of straight lines where half of the facets contain a ring, resembling a chessboard. This pattern, referred to as ring-connected straight lines (RCSLs), is designed to optimize routing efficiency.

The necessary framework has been introduced, and the effects of architecture on routing are studied in the next section.

5. COMPARISON OF DIFFERENT TILE SHAPES

In this section, the impact of tile shapes is analyzed. This effect is two-fold. First, the impact on an individual commodity is studied. For example, not all mesh architectures can always realize a path between two ports. To combat this issue, we provide a modification to the relevant architectures. Second, we introduce metrics that express how well a tile shape can route many commodities at the same time. These are compared for the different tile shapes and trade-offs are discussed.

A. Effect on Single Commodities

None of square tiling, ring-connected straight lines, and triangles can always realize a path between two ports. For example, in Fig. 2(b) once the signal has entered the mesh through one of the ports located at (1,0), it can never exit the mesh through the ports located at (–1,1), (–1,3), (1,4), (3,1), and (3,3). Regardless of the input port, half of the ports are not reachable. This is a systematic problem of these meshes.

We give a proof that this is always the case. Suppose the signal enters through one of the ports located at (1,0). Now, the signal can either enter the square tile that is left adjacent in a counterclockwise direction, or the right adjacent tile in a clockwise direction. When the signal then continues to a neighboring tile, it always reverses the rotation it had before, i.e., from clockwise to counterclockwise or vice versa. This rotation-reversion happens each time the signal enters another tile. In total, this produces a checkerboard-like partition, where half of the squares can only be traversed in a clockwise direction, and the other half of the squares only in a counterclockwise direction as indicated by the red arrows. Now, we can see that the port at (1,4) is unreachable, as this would require reaching either square A in a counterclockwise direction, or to reach B in a clockwise direction. This is impossible when following the checkerboard pattern of rotations. A similar reasoning can be applied to both triangles and RCSLs. This always causes half of the ports to be unreachable for a fixed input port.

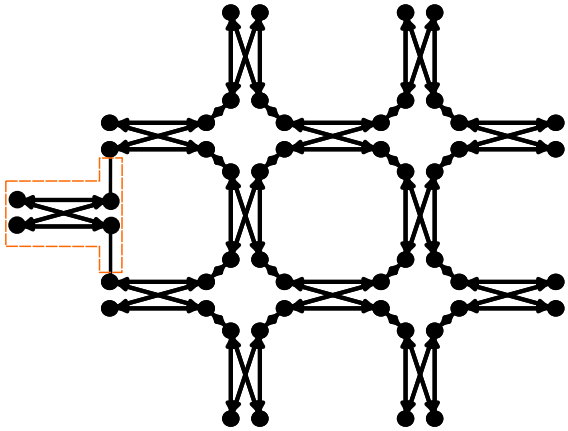

Fig. 4. Indicated structure adds an additional layer. If this is added to the entire outer layer, every two ports can be connected by a path, overcoming a serious limitation of the square/triangle tile shape and RCSL layout.

This is in stark contrast to hexagonal tiling, where between every two ports a possible path exists.

This limitation could be remedied by slightly changing the outer structure, and adding one additional layer of couplers, as seen in Fig. 4. Looking at this additional layer, a light signal can enter any tile in either rotation. This alleviates the limitation. In Ref. [13], the authors provide an additional outer layer as well, but this does not resolve the aforementioned problem. In the rest of this paper, we do not use this additional layer to adhere to the majority of other papers on this topic.

As mentioned before in Section 4, in a mesh with triangular tiles, every path zigzags rather inefficiently in the mesh. Similarly with square tiles, the paths that signals take often seem indirect and wasteful. This idea is captured with the following calculation, which results in a ratio where lower value means more efficient routing.

We want to express the physical distance between a source and a destination, expressed in how many couplers are needed to cover this physical distance. For this, we assume that all couplers have a length of one, and everything else has a length of zero in the mesh. This is similar to introducing a coordinate system for each mesh. Now, we calculate the distance between ports using Pythagoras' theorem. For example, in Fig. 6, the distance between the ports at (0,0) and (0,4) is $\sqrt{(0-0)^2+(0-4)^2}=4$. The distance between (0,0) and (3,3) is $\sqrt{(3-0)^2+(3-0)^2}=\sqrt{18}$. We compare this distance with the actual shortest path in the mesh. Between (0,0) and (3,3), the shortest path contains six couplers. Now we get the desired ratio by dividing the shortest path length by the distance between these two ports, or $\frac{6}{\sqrt{18}}=\sqrt{2}$. For the pair (0,0) and (0,4), the shortest path length is four, and the distance is four as well; hence the ratio for this pair is $\frac{4}{4}=1$. This coincides with the intuitive idea that the shortest path between (0,0) and (0,4) is really efficient, as it goes in a direct line from source to destination. On the other hand, the shortest path from (0,0) to (3,3) goes a bit more indirect. This translates into a higher ratio. We can repeat this process of introducing a coordinate system and calculating the distance

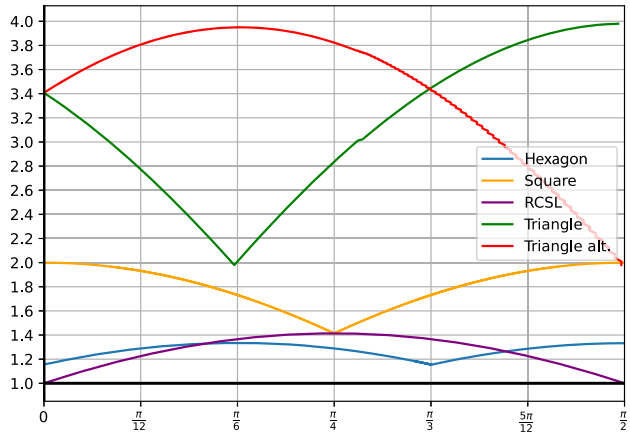

Fig. 5. Ratio of the actual path length compared to a straight line from source to destination. On the x-axis is the relative position of the source and destination node.

between every pair of ports with Pythagoras' theorem. Then we compare this with the actual shortest path length and calculate the ratio for all tile shapes.

The ratio varies within the mesh in all architectures. It depends on the relative location of the source and destination. We define this relative location as the angle between the following two lines. For a source port on either the north or the south side of the mesh, draw a vertical line through this port. Draw the second line through the source and destination port. The relative location is now the angle between these two lines. For a port on the west or east side of the mesh, the first line is horizontal instead. Going back to the example, we calculate the angle between (0,0) and (3,3) in Fig. 6. The first line goes vertically through (0,0) and the second line goes through (0,0) and (3,3). The angle between these two lines is thus 45° or $\frac{\pi}{4}$ radians. In comparison, the angle for the ports (0,0) and (0,4) is 0° or 0 radians as well, as both lines coincide.

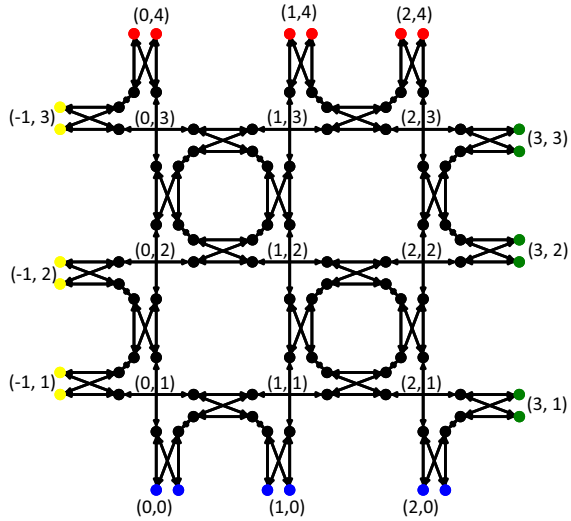

Fig. 6. Mesh with RCSL tiles and the appropriate coordinate system to easily calculate the analytical distance.

Figure 5 expresses this ratio as a function of the relative position. In an ideal mesh, this ratio is as close as possible to one, no matter how the source and destination are positioned. This data is calculated on a sufficiently large mesh, such that a dense collection of angles is realizable.

It can be seen that RCSL tiling manages to achieve a ratio of one when the angle is 0° and 90°, i.e., 0 and $\frac{\pi}{2}$. It is specifically designed with this in mind. On average, hexagonal tile shapes and RCSL achieve a similar ratio. Both RCSL and hexagonal tiling always provide shorter paths than a mesh with square tile shapes. The triangle tile shape behaves a bit special. Its ratio varies between two and four, and it is not only dependent on the angle towards the destination. The input ports can also be divided into two partitions of the input ports; all the ports inside the partition behave similarly. This phenomenon can be seen in Fig. 3 where the left path has a ratio of four, while the right path has a ratio of two, but there is a very small relative location difference. These two belong to different partitions, hence the two different lines for triangle tiles.

B. Analytical Distance in a Mesh

An analytical formula is provided to calculate the length of the shortest path in a mesh. The length is expressed in terms of the number of couplers in the path. This analytical formula is derived for the tile shapes: square, hexagon, and RCSL. The triangle tile shape is omitted here, given the more complicated nature with many different cases that need to be distinguished.

To derive the analytical formula, it is assumed that the overall contour of the mesh is square, although just a requirement of convexity would suffice. These formulas do not take the restrictions into account that not all ports have a path between them. The formula for square tile shapes and RCSL is exact, whereas for hexagons it is an approximation. A more careful analysis can give an exact formula there as well; see Appendix A.

1. Square Tile Shapes

In this section, the exact analytical distance is described in a mesh with square tiles. We introduce a coordinate system; see Fig. 2(b). The coordinate system mainly matters for the ports on the sides, but it has been extended throughout the mesh. Notice that there are always two ports with the same coordinates. These are reachable by the same coupler; hence these are interchangeable. Let the coordinates of the source port be $s = (s_x, s_y)$ and the coordinates of the destination port be $t = (t_x, t_y)$. Suppose the two ports are now on opposite sides, e.g., the north and south sides. If there is a path between these two ports, then the length of the shortest path is

$$d(s,t) = 2\max(|s_x - t_x|, |s_y - t_y|) - 1.$$

Similarly, for ports on the adjacent side, if there is a path, the shortest path has length $d(s,t) = 2\max(|s_x - t_x|, |s_y - t_y|)$. If the ports are on the same side and there is a path, this distance becomes $2\max(|s_x - t_x|, |s_y - t_y|) + 1$.

2. Ring-Connected Straight Lines

A similar coordinate system as for square tiles is used; see Fig. 6. Given that the source and destination ports have coordinates $s = (s_x, s_y)$ and $t = (t_x, t_y)$, the following formula describes the distance between two ports, if there is a path:

$$d(s,t) = |s_x - t_x| + |s_y - t_y|.$$

If both ports are on the same side, the formula changes slightly to $d(s,t) = |s_x - t_x| + |s_y - t_y| + 2$.

3. Hexagon Tile Shapes

For a mesh with hexagonal tiles, a special coordinate system is introduced. These are the cube coordinates, or equivalently, the axial coordinate system. See Fig. 2(c). A good introduction is given online [25]. On the shortest path between two ports, roughly two couplers are used to traverse every hexagon in between these two ports. The distance is now approximately twice the number of hexagonal tiles between their respective hexagons. In the cubic system, this is equal to the sum of the absolute differences of the first two coordinates. Given the coordinates to be $s = (s_{q_1}, s_{q_2}, s_{q_3})$ and $t = (t_{q_1}, t_{q_2}, t_{q_3})$, the distance between two ports becomes

$$d(s,t) = 2(|s_{q_1} - t_{q_1}| + |s_{q_2} - t_{q_2}|).$$

This is an estimation, which is only off by a constant number of couplers. See Appendix A for an exact formula.

C. Effect on Multi-commodities

1. Approach to Allow a Fair Comparison

In this section, it is outlined how a fair comparison is achieved when looking at the routing capabilities for multi-commodity routing. As mentioned before, couplers are the main cause of energy consumption and insertion loss. It thus makes sense to try to minimize the number of couplers, while maximizing the routing capabilities. Now, how can we have a fair comparison between different tile shapes and how do we measure which tile shape is better?

As outlined before, a square contour is used for all comparisons. As an added benefit, there is an unambiguous north, east, south, and west side of the mesh. We will use this as well. However, a fixed contour comes with a drawback. Given a specific number of couplers, chances are small that a mesh with exactly this number of couplers can be constructed. For example, see Fig. 2(b). This is a 4 by 4 mesh made out of square tiles and has exactly 24 couplers. The next size, 5 by 5, has 40 couplers. For any other value between 24 and 40, no mesh with square tiles and an overall square contour exists. To further complicate the matter, different tile shapes have different values for which a mesh is possible. Hence, there does not exist a quantity of couplers such that there is a mesh for every tile shape with that exact number of couplers. Additionally, even when the number of couplers is similar, the number of ports still greatly differs. Often, there is no one-on-one comparison between the number of ports of two different meshes, even when their size is roughly equal.

The lack of a one-on-one match between ports is a tricky problem. Having more ports seems to be an advantage at first glance, but if these ports cannot be used in a meaningful way, then this is rather wasteful. This difference in ports makes it impossible to run the same problem on two different tiled meshes. If the number of ports is fixed instead such that there is a one-on-one match between the ports of meshes consisting of different tile shapes, then the number of couplers in the two meshes can differ greatly, again raising doubts on how fair this comparison is. To avoid these difficulties, routing problems are defined in such a way that they can easily be solved on different meshes with varying amounts of ports while still staying largely the same. This is done as follows.

Ports are numbered on every side from west to east and from south to north. (West, 3) unambiguously specifies the third port on the west side when counting upwards, from south to north. Instead of exactly defining where a commodity should go, a real number between zero and one is chosen and a side. Every real number and a side are now associated with a single port on that side. Say that there are 10 ports on the west side and the number is 0.32. Then the interval between zero and one is divided into 10 equally sized intervals with the first interval being $[0, \frac{1}{10}]$, the second interval $[\frac{1}{10}, \frac{2}{10}]$, and so on. The interval in which the real number falls is then the corresponding port. In this case, the number 0.32 lies in the fourth interval, [0.3,0.4]; hence this corresponds to the fourth port on the chosen side, counted from south to north side. Now a general problem is defined as a set of source-destination pairs, where both the source and the destination are a side and a real number between zero and one. As mentioned before, the meshes of some tile shapes do not allow to draw a path between every possible pair of ports. After the source port is chosen, the destination port is chosen as the closest port to the real number that is reachable from this input port. The methodology above shows how a general problem can be transformed into a problem on a real mesh.

Suppose the general problem is a single commodity with source (West, 0.05) and destination (East, 0.9). Then no matter the tile shape, this commodity is always converted to a commodity going from the west side roughly in the south corner to the east side roughly in the north corner. The same general problem is translated to similar problems on different architectures. This is the basis for a fair comparison between different tile shapes.

2. Different Types of Problems

Monte Carlo simulations are the primary method of comparison between different tile shapes. The lack of realistic usage data makes it hard to construct representative problems. Being completely random appears then as the best choice. These are constructed by randomly choosing a side and a real number between zero and one for every source and destination of a commodity. The destination side is never chosen to be the same side as the source. When translating a general problem to a mesh with a specific tile shape, small constraints are taken into account, to avoid accidentally choosing unroutable configurations. For example, the same port cannot be chosen twice for different commodities. In total, 2058 random test cases are used and these are solved on meshes between 15 and 532 couplers.

Besides this, some predetermined patterns are also studied. These were chosen upfront, to highlight the difference between random routing, and more structured patterns that might occur as well. For example, it was expected that RCSL would perform very well on problem I, almost by design. These problem types are now explained and can be seen in Fig. 7.

I. Take n evenly spaced ports on the south side and make them the source for n corresponding destination ports on the north side, such that each destination is right above its source.

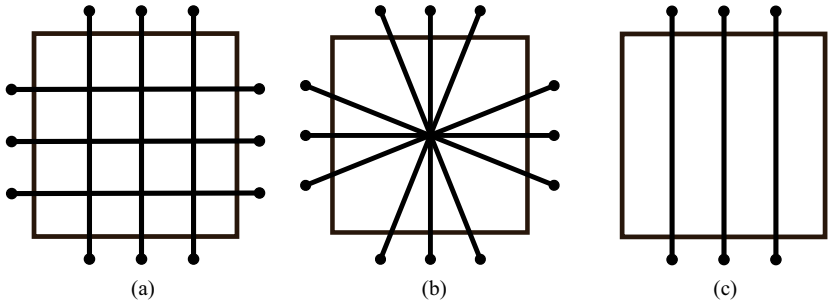

Fig. 7. Simplified representation of the different problem types. (a) Type I, (b) Type II, and (c) Type III.

Now take additional n evenly spaced ports on the west side, and make these the sources of n evenly spaced destination ports on the east side, again such that each destination is right across. We call this Type I.

II. A similar problem to Type I takes the same n evenly spaced ports on opposite sides, but instead for each commodity, if the source port is the nth port counting from the left on the south side, then the destination port becomes the nth port counting from the right on the north side. The ports on the west and east sides are connected similarly. This problem is called Type II.

III. This problem is similar to the first, but only the source and destination ports on the north and south sides are considered; hence there are no commodities that go from east to west. This is Type III.

D. Results and Discussion

Given a fixed problem type and tile shape, the problem is solved for meshes of various sizes with the four studied tile shapes. A solution is calculated by an integer program for smaller problem sizes, and the Aurora algorithm for larger problems, both described in Ref. [20]. The Aurora algorithm iteratively calculates the least-weighted path for every commodity, and by cleverly updating the weights of the various routing resources, discourages multiple commodities from using the same path simultaneously. The integer program states that all routing resources can be used by at most one path. Besides this, other constraints force a path between every source and destination. The objective function is designed to minimize the total number of routing resources. To calculate the maximum number of commodities, a greedy approximation algorithm is used. Given a list of n commodities $[c_1, c_2, \ldots, c_n]$, first $[c_1]$ is routed. The next commodity is added after every successful routing. If at a certain point, $[c_1, \ldots, c_i]$ is routable, but $[c_1, \ldots, c_i, c_{i+1}]$ is not, then c_{i+1} is removed from the list, and c_{i+2} is added instead. At the end of this process, the length of this list is the maximum number of commodities routable at the same time.

We define the following three metrics as expressing how well a mesh can route a problem. These metrics are always interpreted as a function of the number of couplers in the mesh, as couplers are the main source of attenuation and power consumption. As mentioned before, an important caveat is given for RCSL. Many crossings are located in this mesh, which causes additional loss. The advantage or disadvantage of RCSL depends on the relative loss of couplers and crossings. This is discussed later in this paragraph.

1. The maximum number of commodities routable at the same time.
2. The average path length.
3. The shortest path for each commodity, regardless of the other commodities present in the mesh.

In Fig. 8, the average number of routable commodities is displayed. Here a clear trend can be seen, with triangles and squares losing out in favor of RCSL and hexagons. Especially hexagonal tile shapes do well. The problem that there are no meshes with exactly the same number of commodities can be reasonably well taken into account if linear interpolation is used. Using this for a hypothetical mesh of 480 couplers, hexagons could route on average 22.1 commodities, RCSL 18.5, squares 15.0, and triangles 15.7. Thus hexagonal tile shapes can route around 20% more commodities than RCSL. This percentage increases to 40% and 47% more commodities for squares and triangles, respectively. We now look at all meshes between 24 and 480 couplers; there, hexagonal tile shapes can route on average 16% more commodities than RCSL, 39% more commodities than squares, and 49% more than triangles.

These results come with a large caveat that random routing might not be representative, which is why the other three test

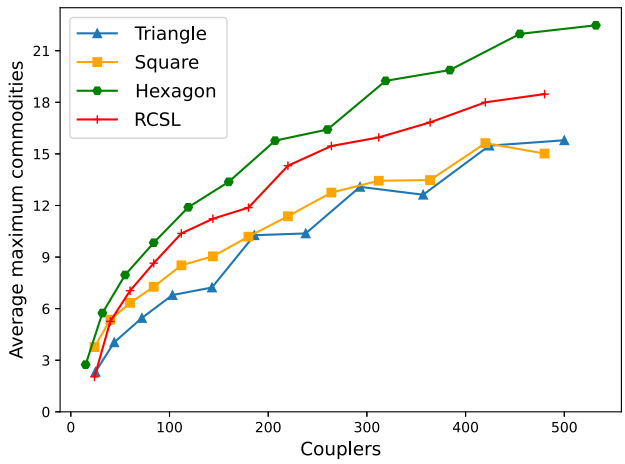

Fig. 8. Average maximum number of commodities routable at the same time.

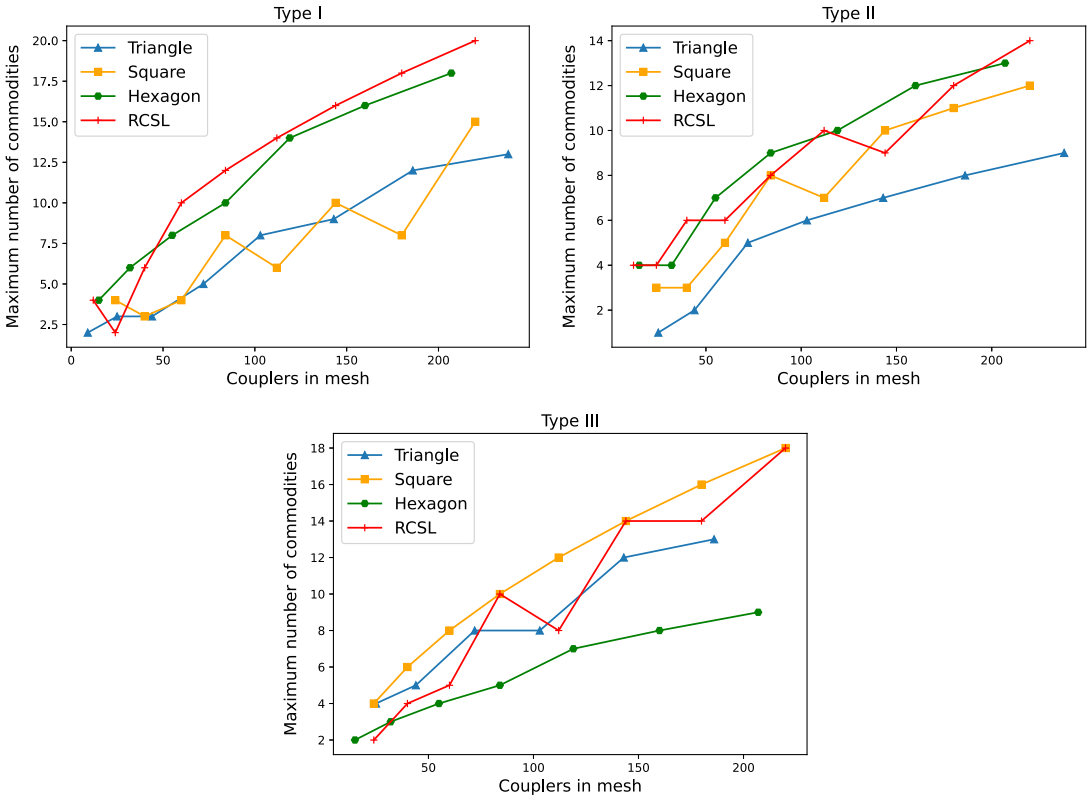

Fig. 9. Maximum number of commodities routable at the same time for the predefined patterns.

set types are introduced. The results of these can be seen in Fig. 9. As expected, RCSL performs well, especially on the Type I problem. Somewhat surprisingly, square tile shapes are the best-performing tile shapes on problem Type III. This can be partially explained by the fact that squares do not use a lot of couplers to form a square, four exactly, and thus a mesh of n couplers contains more squares than, for example, an n coupler mesh of hexagons. Now, a square-tile-shaped mesh is wider than a similarly sized hexagonal-tile-shaped mesh. It can easily route many commodities from the north side to the south side, by configuring all vertical couplers in bar mode, while all horizontal couplers are chosen to be cross mode. This in turn leads to many commodities that have a path, and a high maximum number of commodities.

In Fig. 10, the average length of a path in the solution and the average length of the shortest path are graphed. Here, RCSL does well, and always offers shorter paths. This is not surprising, given the fact that this mesh was specifically designed to have short paths. As predicted, triangular tiles produce especially long paths. Square and hexagon tiles give mostly similar paths. Using interpolation again, on a hypothetical mesh of 480 couplers, the average length of the path in hexagonal tiles is 46.9 couplers, in RCSL this is 36.8, for square tiles 47.5, and for triangles 65.3. Using hexagonal tiles as the base reference again, RCSL has on average 22% shorter paths. The paths in square-shaped tiles are roughly the same, while in triangular tiles these are 39% longer.

At first glance, it is surprising that square and hexagonal tiles give about the same average length given the unequal shortest

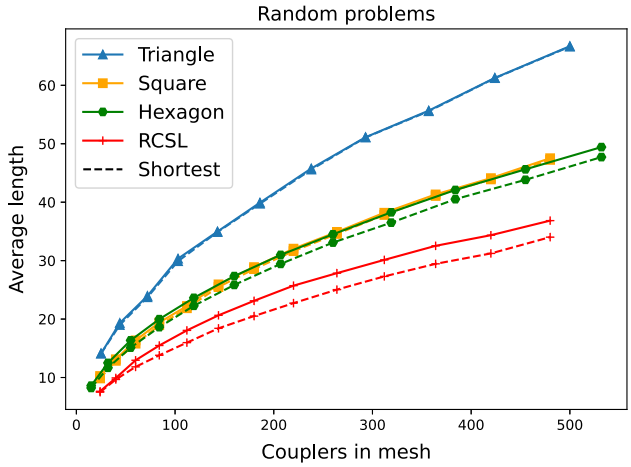

Fig. 10. Average length of a path.

length discussed in Section 5.A. We offer an explanation that is two-fold. First, in a hexagonal-tile-shaped mesh, there are more commodities routed. This increases the chance of detours and thus an increase in path length. Second, notice that the two ports of a commodity lie physically further away in a hexagonal-tile-shaped mesh because hexagons take up more space than squares even if you adjust for the number of couplers in the shape, hence, an increased average length.

In Fig. 11 the average path lengths can be seen for the predetermined patterns. The results are largely the same as for the

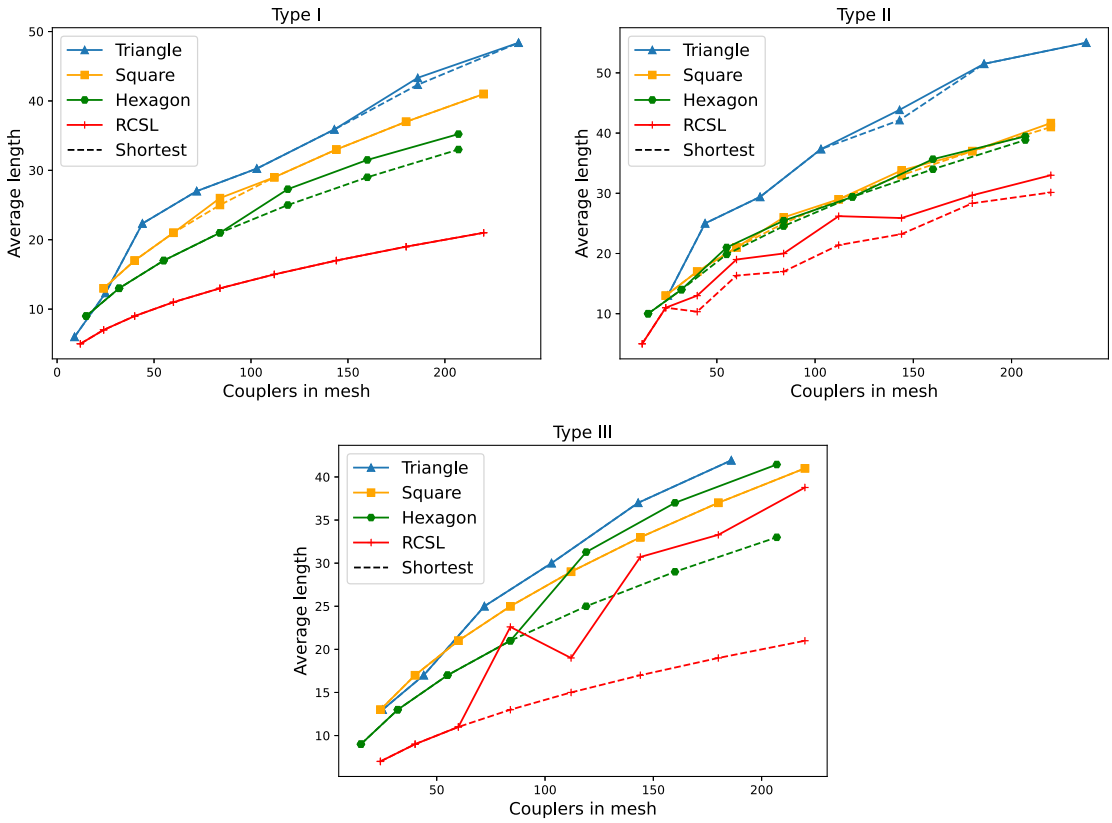

Fig. 11. Average path length in the solution, with the average length of the shortest path marked with a dashed line.

problems with randomly chosen source and destination ports. One remark can be made for the noticeably larger difference between shortest paths and paths in the solution for RCSL in problem Type III. In this problem, RCSL efficiently packs additional paths for commodities by zigzagging as can be seen in Fig. 12. This explains this phenomenon, as these zigzagging paths are a lot longer than their shortest paths.

In conclusion, looking at routing capabilities, it is clear that a preference should be given to hexagonal tile shapes and RCSL over triangular and square tile shapes. Although square tiles performed well for some predefined patterns, in more complex routing schemes, the routing capabilities seem to be inferior. Hexagonal tile shape and RCSL are superior in the maximum number of commodities routed, and RCSL can also boast shorter paths, both for single commodities and while routing multiple commodities. Hexagons especially shine when the routing problem is complex with many paths having to cross each other.

If the mesh is not expected to be densely used and the loss of crossings is around 25% compared to that of couplers, then ring-connected straight lines should be chosen. Otherwise, hexagonal tile shapes are the best performers in densely used meshes. Currently, internal figures estimate these crossings to add around 0.2 dB additional loss. Compared to the insertion loss of a coupler of 0.48 dB as in Ref. [10], this is around 40%. This would make RCSL less interesting as an option. However, it is hard to estimate how the insertion loss of both these structures will evolve. In Ref. [26], the authors demonstrate a crossing with a loss of 0.03 dB. This is sufficient to make crossing loss almost negligible. Crossings are passive as well; hence RCSL has a strong advantage over hexagonal tile shapes when looking at power consumption.

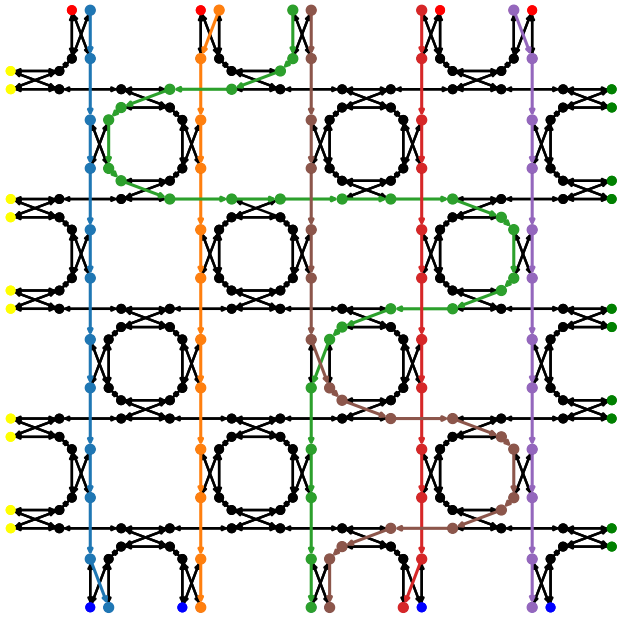

Fig. 12. By smartly routing the green and brown commodities, one additional commodity can be routed.

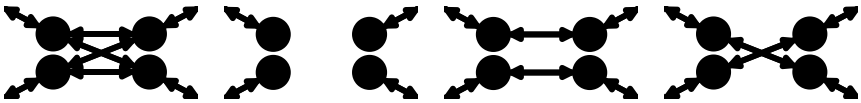

Fig. 13. Comparison of the different ways a coupler can break. (a) Normal, (b) complete failure, (c) only bar mode, and (d) only cross mode.

6. IMPACT OF DAMAGE IN MESH

In this section, the impact of the failure of couplers on the mesh performance is analyzed. Different types of failures are discussed and their impact on the routability of the mesh. Three types of failures are considered: one where the coupler completely fails, and the attenuation in the coupler is 100% for all possible paths; the other two types are where the coupler's driving mechanism fails, and the coupler gets stuck in either cross or bar mode. In Fig. 13, the different types of failure can be seen. A fourth option is that the coupler is stuck in partial coupling mode. If this coupler is used, then this causes a severe degradation of the signal strength, as well as an additional unwanted signal that needs to be transported to an output port. One could try to recombine both these signals again, but interference can appear, again degrading the signal strength for some wavelengths. This would both use up many additional routing resources, and complicate routing. To avoid these issues, we treat couplers stuck in partial coupling mode as completely failed. Depending on the actual implementation of the couplers, certain types of failure are more likely than other types. For example, when the coupler has a heat actuator as a driving mechanism, it can get stuck in cross mode when the heater fails. If the driving mechanism of a microelectromechanical coupler [27] fails, then this causes a coupler to be stuck in bar mode. The combination of both bar and cross mode failure for the different coupler devices using the same implementation seems to be less likely for current coupler implementations but it is included nonetheless.

A. Routability Impact

Given a problem instance that is known to be feasible, i.e., that has a known legal solution, the influence of damage is studied. A percentage of couplers are designated as damaged. The level of damage ranges from 0% to 80% of couplers that are damaged. These damaged couplers are then assigned a damage type in several ways:

1. all couplers are stuck in cross mode;
2. all are stuck in bar mode;
3. all have completely failed;
4. half are stuck in cross mode, half in bar mode;
5. half are stuck in cross mode, and half have completely failed;
6. one-third are stuck in cross mode, one-third in bar mode, and one-third have completely failed.

Once the damage is assigned and a problem is chosen, two quantities are measured. For every problem, the first quantity is the percentage of commodities that still have a path, independent of other commodities. The second quantity is then the percentage of commodities still routable at the same time. To find the maximum number of commodities still routable efficiently, the same greedy algorithm is employed as in Section 5.D; a list is constructed of routable commodities and when a commodity is added that leads to unroutability, it is replaced by the next commodity.

One last distinction is made for problems that use a mesh more "densely". On the same mesh, a problem that consists of many commodities is more severely impacted by failure than a mesh with few commodities. This is logical because there were already few routing resources available, hence little space to reroute affected paths. In Appendix B, we define minimum routing resources ratio, which is the metric used to measure this density. The 25% of problems that score the highest on this metric, is contained in "Simultaneous—Dense". "Simultaneous" is the average of all problems.

In total, 1203 different random feasible problems with varying numbers of commodities were used in this test. The tile shape is always chosen to be hexagonal, given that this is the best-performing tile shape. Given that the other shapes are less flexible routing-wise, the impact of damage will likely only be exacerbated.

B. Results and Discussion

Different types of failures have different effects. As can be seen in Fig. 14(a), if damaged couplers are stuck in cross mode, the effects are rather limited, as long as less than 50% of couplers are damaged. Once three in four couplers are damaged, then the overall performance declines rapidly as well. Compare this to the effects in Fig. 14(b), where the loss of routability occurs more rapidly, with a 50% loss in routing capabilities occurring when only 40% of couplers are damaged. When couplers completely fail, as can be seen in Fig. 14(c), only a 15% loss of couplers already leads to a 50% loss in routing capabilities. If the damage is a mix of different types, then the effects are also similar to a combination of the effects of the damages separately.

In conclusion, the effects of damage greatly depend on the type of damage. Whereas couplers stuck in cross mode still allow for a great deal of routability, complete failure of couplers quickly impedes the ability of the mesh to route anything at all. If couplers fail in cross mode, then a 30% to 40% failure rate is still mostly acceptable and leads to a modest loss in routability of single-digit percentages. On the other hand, when 40% of the couplers fail completely, almost all routing functionality of the mesh is completely gone, and close to 0% of all commodities can still be routed. When couplers are stuck in bar mode, the effect is less pronounced as a complete failure, but it still seriously affects the routing ability of the mesh. A 30% loss of couplers equates to around a 30% loss in routability. While these meshes are partially resilient against couplers that are stuck in cross or bar mode, a complete failure of couplers immediately leads to severe losses in routing capabilities, and

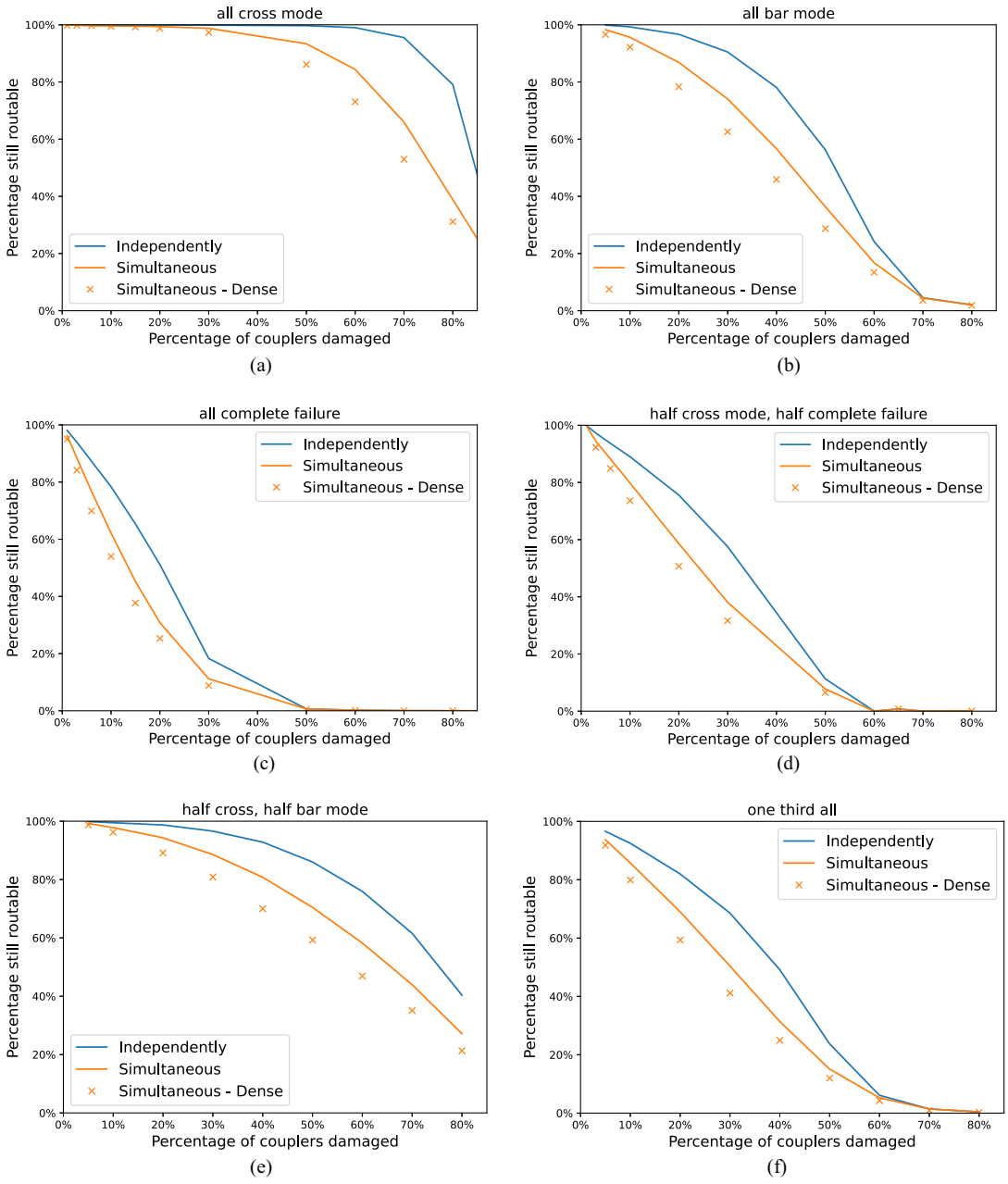

Fig. 14. Impact of different types of coupler failure. Simultaneous—Dense is a subset of problems where there are many different paths and thus the mesh is densely used. (a) Damaged couplers are stuck in cross mode. (b) Damaged couplers are stuck in bar mode. (c) Damaged couplers have failed completely. (d) Half are stuck in cross mode and half have failed completely. (e) Damaged couplers are split in half cross mode and half bar mode. (f) One-third in cross mode, bar mode, and complete failure.

should be avoided as much as possible. This leads to the conclusion that from a routing perspective, coupler driving mechanisms that fail in cross mode as opposed to bar mode or complete failure are preferable.

C. Damage Compensation

These charts can provide a general indication of how much larger a mesh should be, to provide sufficient redundancy in case of coupler failure. Supposing that it is calculated with a perfect mesh, the photonic chip would need 500 couplers to fulfill the required routing. Suppose that the fabrication process is known to have a yield of 90%; thus 10% of couplers are imperfect. Of that 10%, about half of these have completely failed, while the other half are stuck in cross mode. Looking at Fig. 14(d), at 10% damage, around 80% of the routing capabilities are still present. Hence this warrants an increase of the mesh with $\frac{2}{8} = 25\%$. If the mesh is expected to be used densely, the percentage still routable is around 75%. This can be compensated for by multiplying our expected size by the multiplicative inverse of 75%, which is $\frac{4}{3} = 133\%$. This reasoning is

strongly dependent on the fact that the coupler yield does not change depending on the mesh size and that the routing is approximately random.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have studied the effect of different tile shapes on the routing capabilities of photonic meshes. We have shown that hexagonal tile shapes provide the best routing capabilities when the source and destination ports are randomly distributed among the outside of the mesh. Additionally, hexagonal tiles can maintain sufficiently short connections. When a mesh is not expected to be used as densely, our newly introduced design of ring-connected straight lines provides short paths, both from a theoretical perspective and in Monte Carlo simulations.

Furthermore, we have studied the influence of damage on meshes. We have shown that the type of damage matters greatly in the effect it has on the routing capabilities. Once the yield and damage mechanisms are known, an increase in the mesh size can provide redundancy to keep the mesh working as expected.

This work solely focuses on the difference in routing capabilities. A comparable analysis where filter synthesis and placement are taken into account could be insightful once more mature algorithms are available. Instead of meshes with regular tile shapes, irregular meshes might provide added benefit in this case.

APPENDIX A: EXACT ANALYTICAL DISTANCE FORMULA FOR HEXAGONAL TILES

This section gives an exact formula for the analytical distance between two ports in a mesh with hexagonal tiles. This formula is only provided for a fixed source port, but a proper rotation and movement of at most two couplers can immediately bring the port to the same position as the source port that is now chosen.

A schematic representation of a hexagonal mesh is used here. A coordinate system is introduced, where the width is described in the length of a coupler, whereas the height is described in $\frac{\sqrt{3}}{2}$ of the length of a coupler. This is the same as the height that a non-horizontal coupler traverses. This coordinate system is indicated in Fig. 15. For example, the position indicated with A has coordinates (1.5,1). In the following explanation, the coordinates of the destination port (x, y) are assumed to be in the quadrant where both $x, y \geq 0$. The derived formula is valid for all quadrants, as long as one takes the absolute values of the coordinates $|x|$ and $|y|$.

Now, all ports have two classes of possible values for their x-coordinate. This is either $x = 1.5k$ or $x = 1.5k + 0.5$ for a $k \in \mathbb{N}$. Notice that the shortest path to all ports on and under the dashed line consists of only taking couplers to the left. These are studied first. Here, the y-coordinate does not matter. Now, every two subsequent couplers taken increases the x-coordinate by 1.5, vice versa, if $x = 1.5k$; then exactly $2k = \frac{4x}{3}$ couplers are needed. If x is not a multiple of 1.5, then the distance to the start of the coupler of that port is calculated, which has an x-value of $x - 0.5$. From there, the length is increased by one. This is summarized with the following:

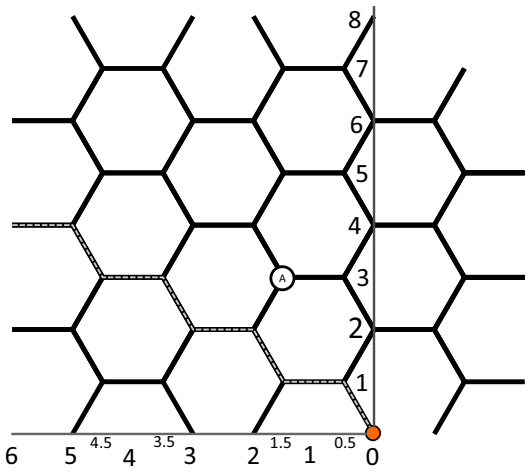

Fig. 15. Coordinate system used to describe an exact analytical distance.

- if $x = 1.5k$, then the distance is $\frac{4x}{3}$;
- if $x = 1.5k + 0.5$, then the distance is $\frac{4(x-0.5)}{3} + 1$.

One port under the dashed line cannot be reached by only taking couplers to the left; this is the port at (3,0), or more generally all ports at $(3k, 0)$ for $k \in N$. This can be resolved by calculating the path to the coordinates of the beginning of the coupler that forms this port, in this case (3.5,1). This length plus one is then the total length to (3,0).

We now turn our attention to destinations above the dashed line. All of these have a shortest path that can be decomposed into two parts: the part that first follows the dashed line until vertically below the destination port; then, the second part goes up by taking alternating couplers that go upwards left and right. The previous formula can be used for the first part. The length of the second part is the difference in y-values between the start and end of the second part. The y-coordinate of the beginning of this part can again be derived from the previous formula. The y-coordinate of the dashed line is expressible in its x coordinate. For every two couplers taken on the dashed line, the height is increased by exactly one, i.e., if $x = 1.5k$, then $y = \frac{3x}{2}$. If $x = 1.5k + 0.5$, then $y = \frac{2(x-0.5)}{3} + 1$. We summarize this with the following cases:

- if $x = 1.5k$, then the distance is $\frac{4x}{3} + \max(0, y - \frac{2x}{3})$;
- if $x = 1.5k + 0.5$, then the distance is $\frac{4(x-0.5)}{3} + 1 + \max(0, y - \frac{2(x-0.5)}{3} - 1)$.

This expresses the distance to all ports. The first term is the calculation for the path on or below the dashed line. The second part, i.e., $\max(0, y - \frac{2x}{3})$, gives the path that purely goes upwards if there is such a path needed in the first place. If not, $y - \frac{2x}{3}$ is negative; hence $\max(0, y - \frac{2x}{3})$ is equal to zero. If $x = 1.5k$, then the calculations are similar, but with an additional correction term applied.

APPENDIX B: DENSELY USED MESHES

In this section, we define when a mesh is densely used with respect to a problem. For this, we introduce the concept of minimum routing resources ratio, which expresses the

percentage of the routing resources that have to be used in any solution to the problem.

Definition 1 (minimum routing resources ratio). Given a mesh and a problem instance C, compute for each commodity c of the problem instance its shortest path irrespective of the other commodities and call this l_c. Denote the total amount of routing resources in the mesh with r_{tot}. The minimum routing resources ratio is now

$$\frac{\sum_{c \in C} l_c}{r_{\text{tot}}}.$$

Any solution to this problem instance needs, almost by definition, at least this percentage of the routing resources. Now in this paper, we use this value as a proxy for how densely used the mesh is by different problems. In the particular case of Section 6.A, the problems are ranked on their minimum routing resources ratio from low to high, and the upper quartile is classified as densely used.

Funding. European Research Council (725555); Fonds Wetenschappelijk Onderzoek (11O0923N, G020421).

Acknowledgment. The authors would like to thank Yu Zhang for his comments and feedback. ChatGPT was utilized to assist in refining the language of this manuscript.

Disclosures. The authors declare no conflicts of interest.

Data Availability. Data underlying the results presented in this paper are not publicly available at this time but may be obtained from the authors upon reasonable request.

REFERENCES

1. R. Soref, "The past, present, and future of silicon photonics," IEEE J. Sel. Top. Quantum Electron. **12**, 1678–1687 (2006).
2. W. Bogaerts, D. Pérez, J. Capmany, *et al.*, "Programmable photonic circuits," Nature **586**, 207–216 (2020).
3. U. L. Andersen, "Photonic chip brings optical quantum computers a step closer," Nature **591**, 40–41 (2021).
4. J. M. Arrazola, V. Bergholm, K. Brádler, *et al.*, "Quantum circuits with many photons on a programmable nanophotonic chip," Nature **591**, 54–60 (2021).
5. D. Marpaung, J. Yao, and J. Capmany, "Integrated microwave photonics," Nat. Photonics **13**, 80–90 (2019).
6. Z. Xie, D. Sánchez-Jácome, L. Torrijos Morán, *et al.*, "Software-defined optical networking applications enabled by programmable integrated photonics," J. Opt. Commun. Netw. **16**, D10–D17 (2024).
7. Y. Shen, N. C. Harris, S. Skirlo, *et al.*, "Deep learning with coherent nanophotonic circuits," Nat. Photonics **11**, 441–446 (2017).
8. G. Genty, "Machine learning and applications in ultrafast photonics," Nat. Photonics **15**, 91–101 (2021).
9. P. L. McMahon, "The physics of optical computing," Nat. Rev. Phys. **5**, 717–734 (2023).
10. D. Pérez-López, A. Gutierrez, D. Sánchez, *et al.*, "General-purpose programmable photonic processor for advanced radiofrequency applications," Nat. Commun. **15**, 1563 (2024).
11. D. Pérez López, "Programmable integrated silicon photonics waveguide meshes: optimized designs and control algorithms," IEEE J. Sel. Top. Quantum Electron. **26**, 8301312 (2020).
12. J. Capmany and D. Pérez, *Programmable Integrated Photonics* (Oxford University, 2020).
13. Z. Gao, X. Chen, Z. Zhang, *et al.*, "Provable routing analysis of programmable photonics," arXiv, arXiv:2306.12607 (2023).
14. L. Zhuang, C. G. H. Roeloffzen, M. Hoekman, *et al.*, "Programmable photonic signal processor chip for radiofrequency applications," Optica **2**, 854–859 (2015).
15. Z. Gao, X. Chen, Z. Zhang, *et al.*, "Automatic synthesis of light-processing functions for programmable photonics: theory and realization," Photonics Res. **11**, 643–658 (2023).
16. W. Bogaerts and L. Chrostowski, "Silicon photonics circuit design: methods, tools and challenges," Laser Photonics Rev. **12**, 1700237 (2018).
17. R. Hamerly, S. Bandyopadhyay, and D. Englund, "Asymptotically fault-tolerant programmable photonics," Nat. Commun. **13**, 6831 (2022).
18. D. A. B. Miller, "Perfect optics with imperfect components," Optica **2**, 747–750 (2015).
19. I. Zand and W. Bogaerts, "Effects of coupling and phase imperfections in programmable photonic hexagonal waveguide meshes," Photonics Res. **8**, 211–218 (2020).
20. F. Vanden Kerchove, X. Chen, D. Colle, *et al.*, "An automated router with optical resource adaptation," J. Lightwave Technol. **41**, 5807–5819 (2023).
21. X. Chen, P. Stroobant, M. Pickavet, *et al.*, "Graph representations for programmable photonic circuits," J. Lightwave Technol. **38**, 4009–4018 (2020).
22. A. López, D. Pérez, P. DasMahapatra, *et al.*, "Auto-routing algorithm for field-programmable photonic gate arrays," Opt. Express **28**, 737–752 (2020).
23. D. Pérez, I. Gasulla, J. Capmany, *et al.*, "Reconfigurable lattice mesh designs for programmable photonic processors," Opt. Express **24**, 12093–12106 (2016).
24. A. Jose, G. Patounakis, and K. Shepard, "Pulsed current-mode signaling for nearly speed-of-light intrachip communication," IEEE J. Solid-State Circuits **41**, 772–780 (2006).
25. https://www.redblobgames.com/grids/hexagons/.
26. Y. Ma, Y. Zhang, S. Yang, *et al.*, "Ultralow loss single layer submicron silicon waveguide crossing for SOI optical interconnect," Opt. Express **21**, 29374–29382 (2013).
27. N. Quack, A. Y. Takabayashi, H. Sattari, *et al.*, "Integrated silicon photonic MEMS," Microsyst. Nanoeng. **9**, 27 (2023).