HiRy: An Advanced Theory on Design of Deadlock-free Adaptive Routing for Arbitrary Topologies

Ryuta Kawano[1], Ryota Yasudo[1], Hiroki Matsutani[1], Michihiro Koibuchi[2], and Hideharu Amano[1]

[1]Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Japan
blackbus@am.ics.keio.ac.jp

[2]National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan
koibuchi@nii.ac.jp

***Abstract*—Recently proposed irregular networks can reduce the latency for both on-chip and off-chip systems with a large number of computing nodes and thus can improve the performance of parallel application. However, these networks usually suffer from deadlocks in routing packets when using a naive minimal path routing algorithm. To solve this problem, we focus attention on a lately proposed theory that generalizes the turn model to maintain the network performance with deadlock-freedom. The theorems remain a challenge of applying themselves to arbitrary topologies including fully irregular networks. In this paper, we advance the theorems to completely general ones. To apply the idea of the turn model to arbitrary topologies, we introduce a concept of *regions* that define continuous directions of channels on an n-dimensional space. Moreover, we provide a feasible implementation of a deadlock-free routing method based on our advanced theorem. To reduce the latency and the number of required Virtual Channels (VCs) with this method, a heuristic approach is introduced to reduce the number of prohibited turns between channels. Experimental results show that the routing method based on our proposed theorem can improve the network throughput by up to 138 % compared to a conventional deterministic minimal routing method. Moreover, it can reduce the latency by up to 2.9 % compared to another fully adaptive routing method.**

***Keywords*—Interconnection Networks, Deadlock-free Routing Algorithm, High Performance Computing, Irregular Networks, Virtual Channels.**

I. Introduction

To improve the performance of large parallel application, low-latency and high-throughput interconnection networks are essential as well as processing performance of computational nodes [1], [2]. The performance on off-chip interconnection networks is usually dominated with delay in switching fabrics (e.g., about a hundred nano-seconds in Infiniband QDR) rather than in a link and for injection. Therefore, researchers have recently focused attention on low-latency networks with high-radix switches, which can be modeled as small-diameter topologies with large degrees [3]–[5].

Meanwhile, other recent approaches have shown that irregular topologies adopted in inter-switch networks can significantly reduce the end-to-end latency [6]–[8]. Moreover, small-diameter topologies with arbitrary network sizes are known to usually have irregular structure [9]. These networks can contribute to reduce the latency and thus to improve the performance of parallel application not only for off-chip networks but for on-chip inter-core networks with low-radix routers [10], [11].

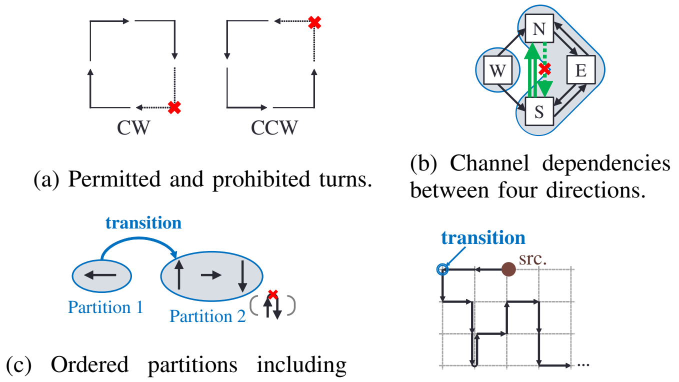
To adopt these networks for practical use, routing algorithms have to guarantee deadlock-freedom in packet transfer. This is because they cannot naively utilize the conventional routing algorithms such as the dimension order routing for k-ary n-cube topologies nor the routing with node labeling for fat-tree topologies. To ensure deadlock-free irregular networks, the channel dependency graph (CDG), derived from the usage of channels with packets, has to be acyclic [12], [13]. To achieve this objective, conventional routing methods often face a trade-off among the required buffer size to implement Virtual Channels (VCs), the achieved latency, and the throughput.

In this work, we propose a novel theorem, called HiRy[1], to design deadlock-free adaptive routing methods for arbitrary network topologies. The theorem is developed from a lately proposed routing theory called EbDa [14], generalization of the turn model [15] for n-dimensional topologies such as Mesh and k-ary n-cube topologies. We introduce a concept of *regions* to define continuous directions of channels to make the idea of the turn model applicable to arbitrary topologies that may include diagonal links.

We also provide a feasible implementation of a topology-agnostic routing algorithm based on our advanced theorem HiRy. Note that this method is one of the possible implementations based on HiRy. Other possible implementations may further improve the network performance compared to our implementation. As the turn model or EbDa, HiRy outlines a routing policy to guarantee deadlock-freedom rather than provides a deadlock-free routing method itself.

The rest of the paper is organized as follows. Sec. II overviews the theorems of EbDa with an example of applying them to a conventional routing method based on the turn model. Sec. III describes our advanced theorem applicable to arbitrary topologies. In Sec. IV, we provide a new deadlock-free routing algorithm for irregular networks based on our theorem. Sec. V shows evaluation of our provided routing method and comparison with conventional routing methodologies. Finally we conclude this paper and mention the future work in Sec. VI.

[1]The name HiRy is derived from the first two letters of the last author's first name and those of the first (or the second) author's.



(a) Permitted and prohibited turns. (b) Channel dependencies between four directions. (c) Ordered partitions including channels available arbitrarily and repeatedly. (d) Example of achieved path.

Fig. 1: An example of Turn model (West-First routing).

II. Related Work

Dally's theory [12] confirms that deadlock-freedom in a network is guaranteed if the channel dependency graph (CDG) induced by the usage of channels with packets is acyclic. Based on this theory, several topology-agnostic deterministic and adaptive routing algorithms have been proposed [16]–[20]. The deadlock-freedom of these algorithms has to be confirmed with an exhaustive cycle dependency check for the topology. This lacks their scalability for an arbitrary network especially with a large network size.

The turn model [15] focuses on directions of channels in n-dimensional Mesh and k-ary n-cube topologies to design deadlock-free adaptive routing for these topologies. Fig. 1a shows an example of the turn model, called the West-First routing. This model shows that any loop is avoided by prohibiting a portion of turns. Various routing methods [21], [22] based on this model can be applied with high scalability because the prohibited turns are independent of the network structure and the network size. However, they still need rigorous proofs that the channel dependency graphs do not include any cycle.

A lately proposed theory, called EbDa [14], also focuses on the channel directions in Mesh and k-ary n-cube topologies to design deadlock-free routing, but verifies the deadlock-freedom without completely relying on Dally's theory. It instead utilizes a partitioning strategy for the channel directions to form an acyclic channel dependency graph. We can illustrate the theorems with the following example of applying them to the West-First routing in a 2D Mesh topology. Let all of the links in the topology be classified according to their directions; i.e., they are grouped into N, S, E, and W links. Fig. 1a shows that the West-First routing prohibits turns from N or S links to W links to break both clockwise and counter-clockwise loops. It is notable that sets of directions {E, N} and {E, S} can be used in arbitrary orders within each set, while {W, N} and {W, S} cannot.

Fig. 1b shows that the directions that packets can use arbitrarily and repeatedly can be arranged into a group, which is called a *partition* in EbDa. In the case of the West-First routing, the two groups {W} and {N, E, S} are generated. The solid arrows in the figure denote the permitted turns between the directions. Moreover, the theorems in EbDa confirm that an additional turn from S to N can be permitted without causing any deadlock. The permitted and prohibited turns between N and S are represented as the doublet and dotted arrows, respectively. A transition between the two partitions in Fig. 1c represents the permitted turns from W to any other direction. The channels in each partition can be used arbitrarily and repeatedly except for the 180-degree turn from N to S in the partition 2.

The deadlock-freedom can be confirmed intuitively with this partitioning model. Fig. 1d shows an example of a path with the routing. Removing loops to avoid deadlock is demonstrated with the following three limitations in routing packets.

1) Packets injected to the source node use the W direction before the turn to the S direction. This turn corresponds to the transition from the partition 1 to the partition 2 in Fig. 1c. After the turn the packets cannot use the W direction again because it means the wrong transition. This uni-directional transition avoids any loop between the partitions {W} and {N, E, S}.
2) After the transition, the packets have to move towards the eastern direction infinitely for the horizontal coordinate axis. It means that there are no loops for the horizontal movement within the latter partition.
3) In addition, the vertical movement of the packets cannot close any loop because of the prohibited 180-degree turn from N to S in the latter partition.

These three limitations for deadlock-freedom correspond to the following three theorems introduced in EbDa, respectively.

1) No cycles are formed with transitions in an ascending order among strictly ordered partitions that do not share any common channel with each other.
2) A partition is loop-free if the number of axes, whose positive and negative directions exist in the partition, is at most one.
3) A partition maintains its deadlock-freedom if the channels, which are parallel with the axis including positive and negative directions mentioned above, are used in a strict order.

It is notable that in the case of the West-First routing in Fig. 1, the vertical channels are ordered as (S, N) to satisfy the condition in the third theorem.

The difference of this work from EbDa is that (1) our theorem HiRy can be applied to arbitrary topologies while the theorems in EbDa cannot, and that (2) we implement a possible routing method based on HiRy and evaluate the performance, while EbDa does not introduce a new routing method. In this work, a concept of *regions* is introduced to define continuous directions of channels. The partitioning strategy is then applied to the regions instead of directions of coordinate axes as in EbDa. We also introduce a heuristic search for our implemented routing method based on HiRy to reduce the latency and the number of required VCs. The

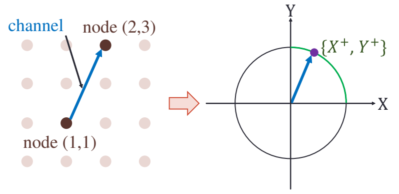


Fig. 2: Channel direction mapped on n-sphere ($n = 2$).

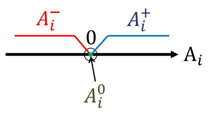
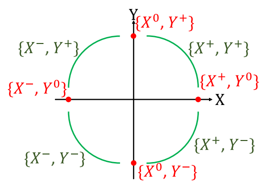
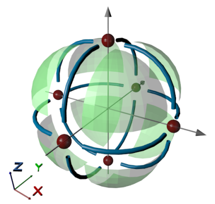


Fig. 3: Negative, zero, and positive coordinates on axis A_i.

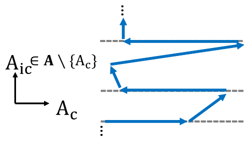


(a) Regions on 2-sphere.

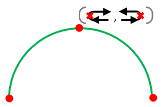


(b) Regions on 3-sphere.

Fig. 4: Regions on 2D- and 3D-surfaces.



(a) Example of achieved path.



(b) Permitted regions for 2D networks with prohibited 180-degree turns between $\{X^-, Y^0\}$ and $\{X^+, Y^0\}$.

Fig. 5: Examples for proof of Lem. 1.

performance is evaluated and compared to that of conventional topology-agnostic routing methods.

III. Theory to Design Deadlock-Free Routing for Arbitrary Topologies

A. Assumptions

Unlike EbDa, channels in the topologies developed on an arbitrary n-dimensional space can be implemented as diagonal ones; that is, they do not have to be arranged in parallel to any of the n coordinate axes. The other assumptions are the same as in EbDa. Wormhole switching networks are assumed while the HiRy theorem also can be applied to virtual cut switching and store-and-forward switching networks. Packets with arbitrary length are routed on the networks. The number of Virtual Channels (VCs) in a physical channel can be an arbitrary positive integer. Moreover, the VCs are treated as disjoint channels even if they are on the same physical channel.

B. Definitions

We introduce an n-sphere[2] centered at the origin in the n-dimensional space and map a direction of a channel to a point on the n-sphere. Fig. 2 shows an example of the mapping. In this example, the direction of the channel from the node $(1, 1)$ to $(2, 3)$ is mapped to the point on the 2-sphere (i.e., the circle) in the first quadrant. Note that the circular arc in the quadrant is labeled as $\{X^+, Y^+\}$. This labeling manner is defined hereinafter.

A coordinate space of an axis A_i is divided into negative, zero, and positive coordinates, which are denoted as A_i^-, A_i^0, and A_i^+, respectively (Fig. 3). By using this division recursively for all axes, the n-dimensional space can be split into 3^n parts. The n-sphere exists on all of the parts except for the origin. Therefore, the following lemma is completed.

Lemma. *An n-sphere can be divided into $3^n - 1$ regions with the division of each axis.*

[2]In this work, an n-sphere is defined as generalization of a circle that overlies an n-dimensional space; e.g., a 2-sphere and a 3-sphere denote a circle and a sphere, respectively.

A *region* in the n-sphere is defined as a set of the coordinates for all axes, where each coordinate is either negative, zero, or positive one that is defined above. For example, the region in the first quadrant on the 2-sphere is described as $\{X^+, Y^+\}$ as shown in Fig. 4a. The 2-sphere can be divided into the following 8 regions.

- 4 regions in one of the quadrants denoted as circular arcs.
- the other 4 regions on the X or Y axis denoted as vertices.

Similarly, as shown in Fig. 4b, the 3-sphere can be divided into the following 26 regions.

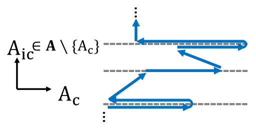
- 8 regions in one of the octants denoted as curved surfaces.
- 12 regions in one of the quadrants on the XY, YZ, or ZX coordinate space denoted as circular arcs.
- 6 regions on the X, Y or Z axis denoted as vertices.

In the same way as EbDa, a *partition* is introduced as a set of channels that packets can use arbitrarily and repeatedly except for 180-degree turns. Note that the 180-degree turn is defined as packet transfer between the two channels that have the opposite directions in the n-dimensional space from each other. In this work, we define the *partition* as a set of the *regions* that represent continuous directions of channels. Unlike EbDa, this definition can treat diagonal links in the n-dimensional space.

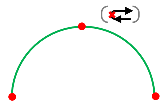
C. Theorem for Deadlock-freedom in Arbitrary Topologies

Lemma 1. *A partition is deadlock-free if the number of axes, whose positive and negative coordinates exist in one of the regions in the partition, is at most one.*

Proof. The deadlock-freedom is supported if the acyclic channel dependency graph (CDG) is formed with packet transfer. Let $\mathbf{A}$ be a set of all axes in the n-dimensional space and $\mathrm{A_c}$ be the axis such that the positive and negative directions can be taken in the partition. For an axis $\mathrm{A_{ic}} \in \mathbf{A} \setminus \{\mathrm{A_c}\}$, packets always have to move in a uni-direction of either the



(a) Example of achieved path.



(b) Permitted regions for 2D networks with prohibited turn from $\{X^-, Y^0\}$ to $\{X^+, Y^0\}$.

Fig. 6: Examples for proof of Lem. 2.

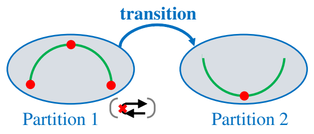
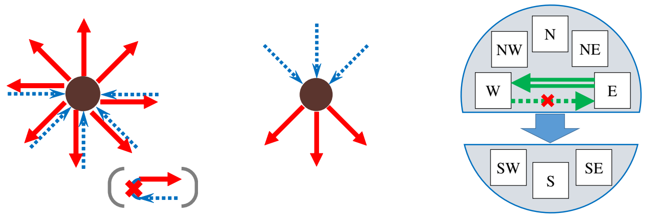


Fig. 7: Ordered partitions including permitted regions of channel direction.



(a) Permitted turns from channels with northern directions.

(b) Permitted turns from channels with southern directions.

(c) Channel dependencies between directions.

Fig. 8: South-Last routing with diagonal links [10].

positive or the negative direction. Therefore, any loop among the channels cannot be formed for these axes. Routing packets along the axis A_c also cannot form any loop because all 180-degree turns are prohibited by default. □

Fig. 5a illustrates an example of an achieved path with a partition that prohibits 180-degree turns. It can avoid loops in the way as described in the proof.

Another example of a partition for a 2-dimensional network, in which the positive and negative coordinates can be taken for the X axis, is shown in Fig. 5b. In this example, the partition includes five regions of $\{X^-, Y^0\}$, $\{X^-, Y^+\}$, $\{X^0, Y^+\}$, $\{X^+, Y^+\}$, and $\{X^+, Y^0\}$. Additionally, it prohibits all 180-degree turns along the X axis (i.e., those between $\{X^-, Y^0\}$ and $\{X^+, Y^0\}$). We can apply the proof of Lem. 1 with replacements of $\mathbf{A} = \{\mathrm{X}, \mathrm{Y}\}$ and $A_c = \mathrm{X}$. Note that the axis A_c is hereinafter referred as a *complete axis*.

Lemma 2. *Let us consider a portion of the channels that are lying on one of the straight lines aligned with the complete axis A_c. If they can be ordered strictly for all of the lines and all packets use the channels in the order within a partition that satisfies the condition of Lem. 1, the partition maintains its deadlock-freedom.*

Proof. When the channels on one of the straight lines parallel with A_c are used in the strict order, we can confirm that the dependencies among them keep acyclic by using topological sort [23]. When this condition is satisfied for all of the lines, this relaxation of the condition in Lem. 1 does not cause any additional loop among all of the channels in the network. □

Fig. 6a illustrates an example of an achieved path with a partition that prohibits a 180-degree turn from A_c^- to A_c^+. The channels parallel to the axis of A_c cannot cause any loop in the use order for the axis $A_{ic} \in \mathbf{A} \setminus \{A_c\}$ because packets have to move in a uni-direction for this axis. They also cannot cause any loop in the use order along the axis of A_c because of the prohibited 180-degree turn.

Fig. 6b shows an example of a partition for a 2-dimensional network, whose configuration is the same as in Fig. 5b except for permitting a turn from $\{X^+, Y^0\}$ to $\{X^-, Y^0\}$. Lem. 2 can be applied to this partition and therefore it keeps its deadlock-freedom. Although we consider networks with a single VC for each physical channel in the above examples, the theorem can also be applied to networks with multiple VCs for each physical channel by ordering the virtual channels and following the order in the packet transfer.

Theorem (HiRy). *Let us assume a set of partitions that do not share any common region with each other. Additionally, we assume that each partition satisfies the condition in Lem. 1 and any 180-degree turn in each partition is achieved following the condition in Lem. 2. If these partitions are ordered strictly and the channels of the partitions are used in the order, the deadlock-freedom is guaranteed.*

Proof. Dependencies among the partitions do not form any cycle when following the order. Each region in the network belongs to at most one of the partitions. This leads to no loops formed in dependencies among the channels in the different partitions. Since the loop-freedom is supported within each partition with Lem. 1 and Lem. 2, the deadlock-freedom is supported as a whole. □

Fig. 7 shows an example of a transition between disjoint partitions. Since no regions are shared between the partition 1 and 2, any channel is also not shared between them. In the same way as EbDa, the transition among these disjoint partitions does not disturb the deadlock-freedom.

D. Case Study

In this section, we apply the HiRy theorem to a conventional topology-agnostic deadlock-free routing method for on-chip networks [10]. This method called the *South-Last routing* is proposed for networks based on 2D Mesh topologies with randomly connected shortcut links. The routing policy is described as follows.

1) As shown in Fig. 8a, it permits arbitrary turns from the channels with W, NW, N, NE, and E directions except for a 180-degree turn from W to E.

Algorithm 1 Generating partitions and their order

```
Input: Dimension of network $n$, # of Virtual Channels (VCs) $v$,
       Network $G = (N, C)$, satisfying $N \subset \mathbb{R}^n$ and $C \subseteq N^2$
Output: Ordered partitions $\mathcal{P} = (P_1, P_2, \cdots, P_{v \cdot 2^{n-1}})$
  Set of axes $\mathbf{A}$=$\{\mathrm{A}_1, \mathrm{A}_2, \cdots, \mathrm{A}_n\}$
  /* Partition for each VC */
  $\mathbf{A}_\mathrm{c} \leftarrow \{\mathrm{A}_1, \mathrm{A}_2, \cdots, \mathrm{A}_n\}$
  for $1 \leq i \leq v$ do
      if $\mathbf{A}_\mathrm{c} = \phi$ then
          $\mathbf{A}_\mathrm{c} \leftarrow \{\mathrm{A}_1, \mathrm{A}_2, \cdots, \mathrm{A}_n\}$
      end if
      Randomly pick $\mathrm{A}_\mathrm{c}$ out of $\mathbf{A}_\mathrm{c}$
      Generate set of partitions $\mathbf{P}_i$
      with axes $\mathbf{A}$ and complete axis $\mathrm{A}_\mathrm{c}$ given (See Alg. 2)
  end for
  Merge sets of partitions $\{\mathbf{P}_i \mid 1 \leq i \leq v\}$
  into set of partitions $\mathbf{P}$
  Sort partitions $\mathbf{P}$ into $\mathcal{P}$ with network $G$ given (See Alg. 3)
```

2) As shown in Fig. 8b, it permits turns from the channels with SW, S, and SE directions only to the channels with the same three directions.

Although Fig. 8a and 8b show diagonal links forming 45-degree angles from the X or Y axis, they can take arbitrary angles unless they are parallel to one of the axes.

We can use the approach of the HiRy theorem in the following way. As in Sec. II, we can generate two partitions of {W, NW, N, NE, E} and {SW, S, SE} shown in Fig. 8c. The transition is permitted from the former partition to the latter. The 180-degree turn from E to W is permitted, while the turn from W to E is prohibited. As a result, the partitions and their order completely matches those shown in Fig. 7.

IV. Deadlock-Free Adaptive Routing Based on HiRy Theory

In this section, we provide a feasible implementation of a deadlock-free routing method based on the HiRy theorem introduced in Sec. III. The dimension of a network n and the number of VCs v for each physical channel are assumed as given inputs. We also assume the given network $G = (N, C)$ arranged on the n-dimensional space, where N and C represent a set of nodes and a set of uni-directional channels, respectively. Based on our new theorem, partitions each of which contains regions are generated and ordered. Packets are routed with permitted turns provided by the ordered partitions.

A. Overview of the Algorithm to Generate Ordered Partitions

Alg. 1 shows the main algorithm to generate ordered partitions, which consists of the following two parts.

1) Generating partitions: The partitions are generated for each VC (See Sec. IV-B) to put VCs in the same physical channel into the different partitions from each other, which can generate a maximum number of partitions. We also set the complete axis A_c for each VC to put as many regions in each partition as possible. The complete axis A_c is changed for each VC to provide variation in the generated partitions for all VCs and thus to achieve better optimization in the sorting part shown below.

Algorithm 2 Generating Partitions in a VC

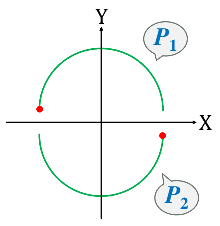
```
Input: Axes $\mathbf{A} = \{\mathrm{A}_1, \mathrm{A}_2, \cdots, \mathrm{A}_n\}$ $(n \geq 2)$,
       Complete axis $\mathrm{A}_\mathrm{c} \in \mathbf{A}$
Output: Set of partitions $\mathbf{P} = \{P_1, P_2, \cdots, P_{2^{n-1}}\}$
  Incomplete axes $\mathbf{A}_\mathrm{ic} = \mathbf{A} \setminus \{\mathrm{A}_\mathrm{c}\}$
  Incomplete indices $\mathbf{I}_\mathrm{ic} = \{i \mid \mathrm{A}_i \in \mathbf{A}_\mathrm{ic}\}$
  Set of non-zero coordinates on $\mathrm{A}_i$ $\mathcal{A}_i = \{A_i^-, A_i^+\}$
  Set of empty partitions $\mathbf{P} = \{P_1, P_2, \cdots, P_{2^{n-1}}\}$
  /* (1) orthant regions */
  $\mathbf{R}_\mathrm{orth} = \{\mathrm{Set}(\mathcal{R}) \mid \mathcal{R} \in \prod_{i \in \mathbf{I}_\mathrm{ic}} \mathcal{A}_i\}$
  for $1 \leq j \leq n$ do
      Pick $R_{\mathrm{orth}_j}$ out of $\mathbf{R}_\mathrm{orth}$
      for $A_\mathrm{c}^* \in \{A_\mathrm{c}^-, A_\mathrm{c}^0, A_\mathrm{c}^+\}$ do
          $P_j$.add($\{A_\mathrm{c}^*\} \cup R_{\mathrm{orth}_j}$)
      end for
  end for
  /* (2) boundary regions */
  for $\mathbf{A}_\mathrm{bnd} \in$ Power($\mathbf{A}_\mathrm{ic}$)$\setminus\{\phi, \mathbf{A}_\mathrm{ic}\}$ do
      $\mathbf{I}_\mathrm{bnd} = \{i \mid \mathrm{A}_i \in \mathbf{A}_\mathrm{bnd}\}$
      $\mathbf{R}_\mathrm{bnd} = \{\mathrm{Set}(\mathcal{R}') \mid \mathcal{R}' \in \prod_{i \in \mathbf{I}_\mathrm{bnd}} \mathcal{A}_i\}$
      for $R_\mathrm{bnd} \in \mathbf{R}_\mathrm{bnd}$ do
          $\mathbf{P}_\mathrm{bnd} = \{P_j \mid R_{\mathrm{orth}_j} \supset R_\mathrm{bnd}\}$
          Randomly select $P_\mathrm{bnd}$ from $\mathbf{P}_\mathrm{bnd}$
          $R_\mathrm{zero} = \{A_i^0 \mid i \in \mathbf{I}_\mathrm{ic} \setminus \mathbf{I}_\mathrm{bnd}\}$
          for $A_\mathrm{c}^* \in \{A_\mathrm{c}^-, A_\mathrm{c}^0, A_\mathrm{c}^+\}$ do
              $P_\mathrm{bnd}$.add($\{A_\mathrm{c}^*\} \cup R_\mathrm{bnd} \cup R_\mathrm{zero}$)
          end for
      end for
  end for
  /* (3) regions of uni-directions in complete axis $\mathrm{A}_\mathrm{c}$ */
  Randomly sample ($P_\mathrm{pos}, P_\mathrm{neg}$) from $\mathbf{P}$
  $R_\mathrm{ic\_zero} = \{A_i^0 \mid i \in \mathbf{I}_\mathrm{ic}\}$
  $P_\mathrm{pos}$.add($\{A_\mathrm{c}^+\} \cup R_\mathrm{ic\_zero}$)
  $P_\mathrm{neg}$.add($\{A_\mathrm{c}^-\} \cup R_\mathrm{ic\_zero}$)
```

2) Merging sets of partitions for all VCs and sorting them: We introduce a heuristic approach (See Sec. IV-C) to generate their order that ensures all source-and-destination pairs reachable and that achieves as the small average minimal path length as possible.

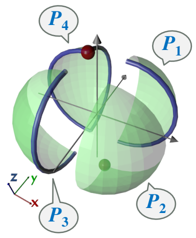
B. Generating Partitions for each VC

Partitions including regions are generated based on the HiRy theorem in Alg. 2. In order to put as many regions into each partition as possible, initially 2^{n-1} partitions are generated. Note that the number 2^{n-1} is derived from the number of orthants in the $(n-1)$-dimensional space constructed with the given axes except for the complete axis A_c. The following three kinds of regions are added to the 2^{n-1} partitions.

1) Orthant regions: For each partition, a region located on the corresponding orthant is added. Note that the region does not contain a zero-coordinate for any of the incomplete axes $\mathbf{A}_\mathrm{ic}$.
2) Boundary regions: They are defined as the regions that multiple orthants are adjacent to in the $(n-1)$-dimensional space. Each of them is added to one of the partitions that include one of the neighboring orthant regions. Note that these regions exceptionally do not include regions of uni-directions in the complete axis A_c, which are described subsequently.



(a) Example of generated partitions for 2D network.



(b) Example of generated partitions for 3D network.

Fig. 9: Generated partitions for a VC.

3) Regions of uni-directions in the complete axis A_c: These two regions are individually added to the different partitions from each other. This is because of the following reason. If these are added to the same partition, we have to limit the use of 180-degree turns along the axis A_c as described in Lem. 2.

In Alg. 2, Power(_) and $\prod$ denote the power set and the Cartesian product of sets, respectively. Moreover, Set(_) is defined as conversion from an ordered tuple to unordered one.

In the example for 2-dimensional networks in Fig. 9a, the following two partitions are generated.

- $P_1 = \{\{X^-, Y^0\}, \{X^-, Y^+\}, \{X^0, Y^+\}, \{X^+, Y^+\}\}$
- $P_2 = \{\{X^+, Y^0\}, \{X^+, Y^-\}, \{X^0, Y^-\}, \{X^-, Y^-\}\}$

In the figure, the following sets of regions are represented as circular arcs.

- $\{\{X^-, Y^+\}, \{X^0, Y^+\}, \{X^+, Y^+\}\}$
- $\{\{X^+, Y^-\}, \{X^0, Y^-\}, \{X^-, Y^-\}\}$

These two sets correspond to (1) the orthant regions; that is, the two partitions for the coordinates of Y^+ and Y^- are generated. In the case of the 2-dimensional space, (2) the boundary regions do not exist and thus they are not added. Furthermore, the regions of $\{X^-, Y^0\}$ and $\{X^+, Y^0\}$ are added to the partitions P_1 and P_2, respectively, as (3) the regions in the complete axis.

Similarly, the following four partitions are generated for 3-dimensional networks as shown in Fig. 9b.

- $P_1 = \{\{X^+, Y^+, Z^-\}, \{X^+, Y^+, Z^0\}, \{X^+, Y^+, Z^+\}, \{X^+, Y^0, Z^-\}, \{X^+, Y^0, Z^0\}, \{X^+, Y^0, Z^+\}\}$
- $P_2 = \{\{X^+, Y^-, Z^-\}, \{X^+, Y^-, Z^0\}, \{X^+, Y^-, Z^+\}, \{X^0, Y^0, Z^-\}\}$
- $P_3 = \{\{X^-, Y^-, Z^-\}, \{X^-, Y^-, Z^0\}, \{X^-, Y^-, Z^+\}, \{X^0, Y^-, Z^-\}, \{X^0, Y^-, Z^0\}, \{X^0, Y^-, Z^+\}\}$
- $P_4 = \{\{X^-, Y^+, Z^-\}, \{X^-, Y^+, Z^0\}, \{X^-, Y^+, Z^+\}, \{X^0, Y^+, Z^-\}, \{X^0, Y^+, Z^0\}, \{X^0, Y^+, Z^+\}, \{X^-, Y^0, Z^-\}, \{X^-, Y^0, Z^0\}, \{X^-, Y^0, Z^+\}, \{X^0, Y^0, Z^+\}\}$

In the figure, (1) the orthant regions are denoted as curved surfaces; e.g., the following orthant regions are added to P_1: $\{X^+, Y^+, Z^-\}, \{X^+, Y^+, Z^0\}, \{X^+, Y^+, Z^+\}$. Furthermore, (2) the boundary regions are denoted as circular arcs; e.g., the following boundary regions are added to P_1: $\{X^+, Y^0, Z^-\}, \{X^+, Y^0, Z^0\}, \{X^+, Y^0, Z^+\}$. Note that they can be alternatively added to the neighboring partitions; e.g., the regions mentioned above can be added to P_2 instead of P_1. (3) The regions of the complete axis are denoted as vertices; e.g., $\{X^0, Y^0, Z^-\}$ is added to P_2.

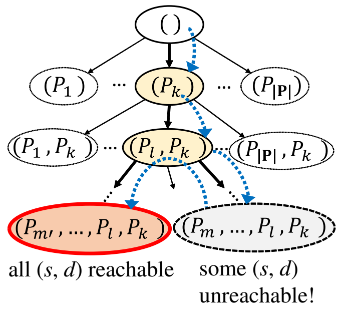


Fig. 10: Best-first search for appropriate order of partitions.

Algorithm 3 Sorting Partitions

```
Input: Set of partitions $\mathbf{P} = \{P_1, P_2, \cdots, P_{|\mathbf{P}|(=v \cdot 2^{n-1})}\}$,
       Network $G = (N, C)$, satisfying $N \subset \mathbb{R}^n$ and $C \subseteq N^2$
Output: Ordered partitions $\mathcal{P} = (P'_1, P'_2, \cdots, P'_{|\mathcal{P}|(=v \cdot 2^{n-1})})$
  MAX_ITERATION ← $1,000$
  Set of temporary partition orders $\mathbf{T} = \{()\}$
  $i_{\text{iter}} \leftarrow 0$
  while $\mathbf{T} \neq \phi$ and $i_{\text{iter}}$ <MAX_ITERATION do
    $i_{\text{iter}} \leftarrow i_{\text{iter}} + 1$
    Pick $T$ out of $\mathbf{T}$ that maximizes # of reachable $(s, d)$-pairs,
    breaking ties with ASPL achieved with $T$ in ascending order
    if All $(s, d)$-pairs in $G$ reachable with $T$ and $|T| = |\mathbf{P}|$ then
      Return $T$ as $\mathcal{P}$
    else
      for all $\{P \mid P \in \mathbf{P} \ \wedge P \notin \text{Set}(T)\}$ do
        $T' =$ copy of $T$
        Insert $P$ to head of $T'$
        $\mathbf{T}$.add($T'$)
      end for
    end if
  end while
  /* No valid partition order found with iterative search */
  Raise ERROR
```

C. Sorting Partitions

After generating the sets of partitions for all VCs, they are merged and sorted for the given network G. Alg. 3 shows our implementation of sorting partitions.

In order to support all (s, d)-pairs in the network reachable, we adopt a heuristic best-first search to find the order of the partitions that ensures all (s, d)-pairs reachable. An example of the search is illustrated in Fig. 10. Each vertex in the tree represents temporary ordered partitions. Initially the tree has only one empty vertex as a root vertex. For each iteration, a visited vertex T is selected among unvisited vertices that maximizes the number of reachable (s, d)-pairs. If there are some ties, they are broken with the average minimal path length in an increasing order. The search is continued by adding children vertices of T for all of the rest partitions. Each child vertex is generated by copying T and inserting each one of the rest partitions to the head of the copy. If the

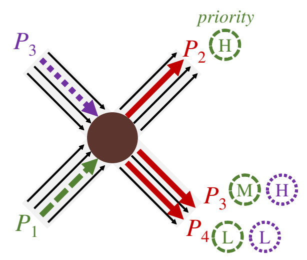


Fig. 11: Output VCs to be requested and their priority.

visited vertex T includes all of the partitions in **P** and there exist some unreachable (s, d)-pairs, the vertex T is discarded and the search is continued by going back to another vertex of the tree. On the other hand, if T includes all of the partitions and ensures all (s, d)-pairs reachable, T is returned as ordered partitions $\mathcal{P}$. In this work the number of the iteration is limited to 1,000 to reduce the computational complexity.

D. Routing Packets

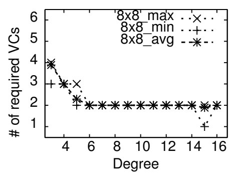
Packets can adaptively use multiple minimal paths that are available with the partitions and their ordering between source and destination nodes. Although they can also use non-minimal paths, we do not recommend using them because they cause degradation in the throughput and the latency depending on the given networks and traffic patterns.

When the packets request allocation for multiple output VCs that induce the minimal paths, the adaptivity can be improved by giving priority to the VCs in the order of the corresponding partitions. Fig. 11 shows an example of requested output VCs and their priority. In this example, the central node has three output VCs inducing minimal paths for a destination node d represented as thick arrows. They belong to the partitions P_2, P_3, and P_4 from the top. Let a packet, whose destination node is d, arrive at the node. When the packet arrives from the input VC that belongs to the partition P_1, represented as the dotted arrow at the lower left, it can use all of the three output VCs according to the partition order. The packet then requests allocation for the three VCs giving higher priority to the partition with a smaller index, as shown in the figure with the characters H (High), M (Middle), and L (Low) enclosed with dotted circles at the right. On the other hand, the packet from the input VC that belongs to the partition P_3, represented as the dotted arrow at the upper left, cannot use the output VC in the partition P_2 because of the partition order. That is why the packet requests for the two VCs in the partitions P_3 and P_4, giving higher priority to that in P_3.

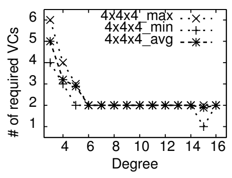
The deadlock-freedom is confirmed as follows. The regions in each partition satisfy the condition in Lem. 1. Moreover, there are no 180-degree turns in each partition that are mentioned in Lem. 2. This is because the two regions of the complete axis are added to the different partitions from each other. In addition, the partitions are ordered and are used in the order to satisfy the condition in the theorem of HiRy.

Algorithm 4 Calculating minimal number of required VCs

```
Input: Dimension of network $n$,
       Network $G = (N, C)$, satisfying $N \subset \mathbb{R}^n$ and $C \subseteq N^2$
Output: Minimal number of required Virtual Channels (VCs) $v_{\min}$
  $v_{\min} \leftarrow 0$
  do
      $v_{\min} \leftarrow v_{\min} + 1$
      Try to generate ordered partitions $\mathcal{P}$
      with $n$, $v_{\min}$, and $G$ given (See Alg. 1)
  while $\mathcal{P}$ ensuring all $(s, d)$-pairs in $G$ reachable not generated
```

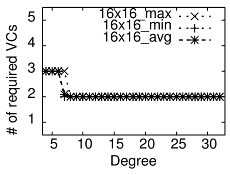


(a) 8×8 random topology.

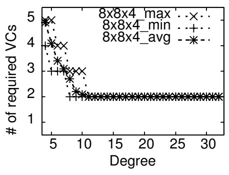


(b) 4×4×4 random topology.

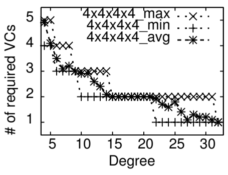
Fig. 12: Number of required Virtual Channels for 64 nodes.



(a) 16×16 random topology.



(b) 8×8×4 random topology.



(c) 4×4×4×4 random topology.

Fig. 13: Number of required Virtual Channels for 256 nodes.

V. Evaluation

In this section, the implemented routing method based on HiRy is evaluated and compared with conventional topology-agnostic routing methods. In this evaluation, routing methods are applied to random regular topologies. The degree of each node is denoted as deg(G); that is, the number of bi-directional links for each node is equal to deg(G). The value deg(G) is the same for all nodes in the network G.

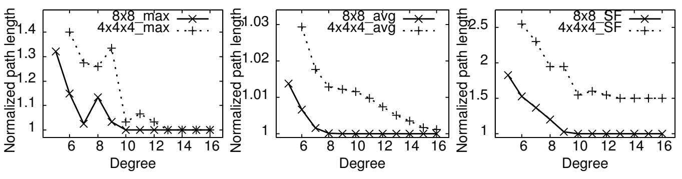
The routing method is applied to 64- and 256-node random regular topologies. These topologies are developed on the following coordinate spaces.

- 64 nodes: 8×8 $(n = 2)$, $4 \times 4 \times 4$ $(n = 3)$
- 256 nodes: 16×16 $(n = 2)$, $8 \times 8 \times 4$ $(n = 3)$, $4 \times 4 \times 4 \times 4$ $(n = 4)$

The nodes are arranged on the lattice positions in each space.

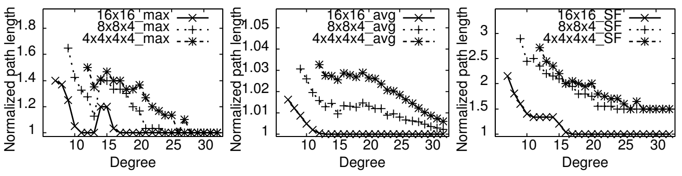
A. Minimal Number of required VCs

In this section, the minimal number of Virtual Channels required to ensure all (s, d)-pairs in G reachable is evaluated. The degree of each node, deg(G), is varied from 3 to 16 for 64 nodes, and from 4 to 32 for 256 nodes. For each ($|N|$,deg(G))-pair, 10 random topologies are generated to get the maximum, minimum, and average numbers of required VCs, where $|N|$ represents the number of nodes. To calculate the minimal number of required VCs, the number of VCs $v_{\min}$ is incremented to achieve Alg. 1 repeatedly until a partition



(a) Maximum. (b) Average. (c) Stretch Factor.

Fig. 14: Normalized path lengths for 64 nodes with 2 VCs.



(a) Maximum. (b) Average. (c) Stretch Factor.

Fig. 15: Normalized path lengths for 256 nodes with 2 VCs.

order $\mathcal{P}$, ensuring all (s, d)-pairs reachable, is generated, as shown in Alg. 4.

For 64 nodes, the maximum numbers of required VCs in the case of deg(G)= 3 are 4 for the 8×8 topology and 6 for the $4 \times 4 \times 4$ topology, as shown in Fig. 12a and 12b, respectively. From these results, it can be seen that when a network with small degrees is developed in a space of a large dimension, it requires relatively many VCs to make all (s, d)-pairs reachable. This is because the large dimension divides the channels in the network into many partitions, each of which includes relatively a small number of available directions of the channels because of the small degree.

Similarly, for 256 nodes, the maximum number of required VCs for the 16×16 topology is 3, while those for the $8 \times 8 \times 4$ and $4 \times 4 \times 4 \times 4$ topologies are both 5, as shown in Fig. 13. Moreover, the minimal numbers of the degree that achieve the number of required VCs equal to 2 is 7, 8 and 10 for the 16×16, $8 \times 8 \times 4$, and $4 \times 4 \times 4 \times 4$ topologies, respectively. On the other hand, the required number of VCs can be equal to 1 in the case of deg(G) ≥ 22 for the $4 \times 4 \times 4 \times 4$ topology. This is because in this case there are relatively many partitions that include a large number of channels, which leads to high reachability of packets.

The proposed method based on HiRy can be applied to completely irregular networks with a moderate value of the dimension n. This is because the increase of n exponentially affects both the increase in the number of partitions and the decrease in the average number of channels in each partition, and thus neutralizes the significant decrease of v and the increase of deg(G). In addition, it is important to select the appropriate dimension n in order to keep a good balance between the number of partitions and the number of channels in each partition to reduce the number of required VCs. To develop a new methodology to get an optimal dimension n is left for our future work.

TABLE I: Network parameters.

Simulation period	100,000 cycles
Packet size	1 flit
Number of VCs	2
Buffer size per VC	8 flits
Number of pipeline stages	4

B. Path Lengths

In this evaluation, the maximum degrees are set to 16 for 64 nodes and 32 for 256 nodes. The number of VCs is fixed to 2. In a similar way to the previous evaluation, 10 random topologies are generated to evaluate the average values of the following three metrics.

- Maximum path length.
- Average path length.
- Stretch Factor, which denotes the maximum factor of the path lengths for all (s, d)-pairs.

The maximum and average path lengths are divided by those of the shortest path length to obtain the normalized values.

For 64 nodes, Fig. 14 shows that the normalized values get close or equal to one when the degrees become large. Moreover, the networks with the small dimension can reduce the values. For the 2-dimensional networks, the shortest path routing can be achieved with 2 VCs in the case of deg(G) ≥ 10. On the other hand, for the 3-dimensional networks, the normalized value only of the maximum can be equal to one in the case of deg(G) ≥ 13. Although the increasing rate in the average is only 3 % for deg(G) = 6, the value of the SF does not fall below the value of 1.5.

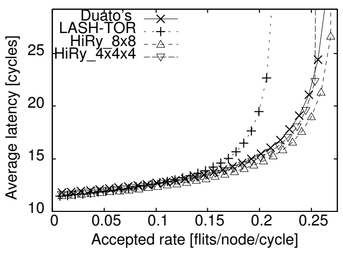
Similarly, for 256 nodes, the 2-dimensional networks can achieve the shortest path routing in the case of deg(G) ≥ 17, while the networks with larger dimensions cannot achieve, as shown in Fig. 15. Nonetheless, the 3- and 4-dimensional networks can suppress the increase rates in the average by 1.6 % and 3.3 %, respectively.

In summary, the configuration of the routing method can be varied depending on the given network or the performance to be achieved. A small-degree network with a small dimension can reduce the number of required VCs with the modestly small average path length. Moreover, a large-degree network with a small dimension has the possibility of achieving the shortest path routing, while that with a large dimension can be implemented with the smaller number of VCs.

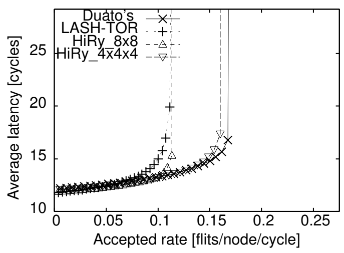
C. Network Simulation

A cycle-accurate network simulator Booksim [24] is used for evaluation. Network parameters for the simulation are shown in Tab. I. In this evaluation, the following three routing methods are compared.

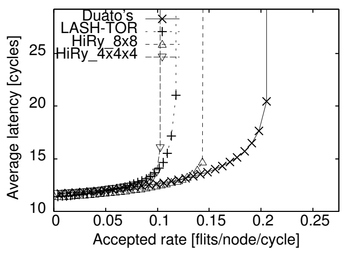
- HiRy: In this section, the legend *HiRy* denotes the implemented routing method described in Sec. IV.
- LASH-TOR [20]: This method splits a path for each (s, d)-pair into sub-paths that are assigned to multiple VCs. In this work, we assume that this method can



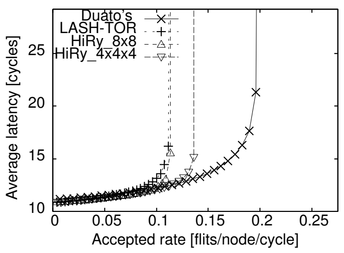
(a) Uniform traffic.



(b) Transpose traffic.

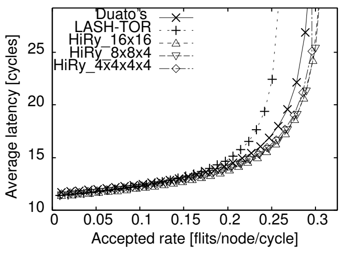


(c) Shuffle traffic.

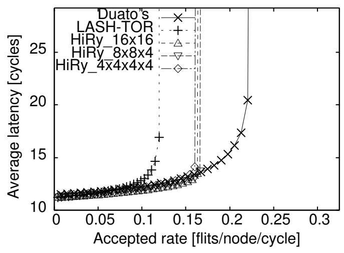


(d) Reverse traffic.

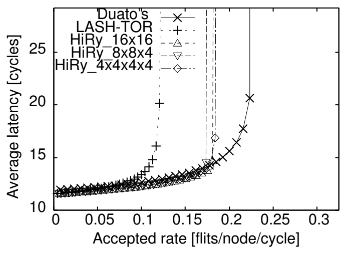
Fig. 16: Network performance for 64 nodes.



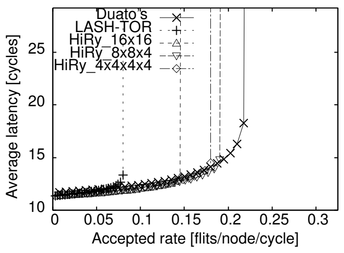
(a) Uniform traffic.



(b) Transpose traffic.



(c) Shuffle traffic.



(d) Reverse traffic.

Fig. 17: Network performance for 256 nodes.

achieve deterministic shortest path routing using transitions among multiple VCs. The routing function is implemented as $R : N \times N \rightarrow C \times V$ that returns the physical output channel c_{out} and an index of the virtual channel v_{out} for the given (s, d)-pair.

- Duato's protocol [18]: This method can achieve minimal adaptive routing with non-minimal escape paths by utilizing multiple VCs. As the escape paths, we adopt the up*/down* routing [17] with the spanning trees optimization method [25].

The degrees of networks are set to 6 for 64 nodes and 13 for 256 nodes. These values are derived from the minimal numbers of degrees that LASH-TOR can achieve the shortest path routing with 2 VCs for the networks with. The simulation is performed under uniform, transpose, shuffle, and reverse traffics [26] for 64 and 256 nodes, as shown in Fig. 16 and 17, respectively.

For 64 nodes, the saturation throughput of HiRy with dimension $n = 3$ is reduced by 4.5 % compared to that of Duato's protocol in the uniform traffic, as shown in Fig. 16a. The similar reduction can be seen in the other traffics except for the shuffle traffic. Additionally, the latency of HiRy with dimension $n = 3$ is increased by 0.3 % compared to that of LASH-TOR. These results stem from the prohibited turns, which induce both the partial adaptivity in routing packets and the non-minimal paths for some (s, d)-pairs. On the other hand, the saturation throughput of HiRy with $n = 2$ and $n = 3$ is both larger than that of LASH-TOR in most of the traffics. This is because LASH-TOR cannot use the alternative paths to reduce congestion, while HiRy can use them by choosing the multiple paths adaptively. In this evaluation, HiRy can increase the saturation throughput by up to 43.2 %. Moreover, HiRy reduces the latency by up to 2.5 % compared to Duato's protocol. This is because Duato's protocol utilizes non-minimal escape paths even when the traffic load is low, which leads to the increased path lengths. By contrast, HiRy achieves the shortest paths for most of the (s, d)-pairs in the same condition.

For 256 nodes, HiRy can improve the performance especially in the synthetic traffic patterns. it can increase the saturation throughput by 138 % in the reverse traffic compared to LASH-TOR, as shown in Fig. 17d. Moreover, it can reduce the latency by up to 2.9 % in the shuffle traffic compared to Duato's protocol, as shown in Fig. 17c.

In summary, the implemented routing method based on HiRy can achieve the network performance comparable to or better than the conventional routing methods for completely irregular networks. It can achieve the high saturation throughput by reducing the number of prohibited turns. Moreover, it can achieve the low latency by using the shortest paths in routing most of the packets in the traffics.

VI. Conclusion and Future Work

In this work, we propose HiRy, the theorem for designing topology-agnostic deadlock-free routing, and provide a feasible implementation of an adaptive routing method for arbitrary networks based on HiRy. HiRy is developed from EbDa, the generalization of the turn model. We advance the theorems by introducing the concept of *regions* that define continuous directions of channels for arbitrary networks. Moreover, the implemented routing method based on HiRy can increase the number of permitted paths and thus can improve the network performance. To support all source-and-destination pairs reachable and to reduce the average path length, a heuristic approach is introduced.

Experimental results show that the routing method based on HiRy can be implemented with only one VC for each physical channel for a 256-node random topology with the degree of 22. Moreover, for a 256-node random topology with the degree of 17, the method can achieve the shortest path routing with two VCs for each physical channel. The results from the network simulation show that it can increase the throughput by up to 138 % compared to LASH-TOR that is one of the deterministic minimal routing methods. Furthermore, it can reduce the latency by up to 2.9 % compared to Duato's protocol that is one of the fully adaptive routing methods.

As a future work, we will focus on developing a new methodology to find the appropriate dimension n for the given network and the number of VCs. Although the dimension n is given for the routing algorithm in this work, the choice of the dimension significantly influences the minimum number of required VCs or the resulted path lengths. We believe that there is still room for more effective utilization of the proposed theorem HiRy.

Acknowledgment A part of this work was supported by JSPS KAKENHI Grant Number 15J03374.

References

[1] K. Scott Hemmert et al, "Report on Institute for Advanced Architectures and Algorithms, Interconnection Networks Workshop 2008," http://ft.ornl.gov/doku/_media/iaaicw/iaa-ic-2008-workshop-report-v09.pdf.

[2] J. Tomkins, "Interconnects: A Buyers Point of View," ACS Workshop, Jun 2007.

[3] J. Kim, W. J. Dally, and D. Abts, "Flattened Butterfly: a Cost-Efficient Topology for High-Radix Networks," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, Jun 2007, pp. 126–137.

[4] W. Bao, B. Fu, M. Chen, and L. Zhang, "A High-Performance and Cost-Efficient Interconnection Network for High-Density Servers," in *Proc. of the IEEE International Conference on High Performance Computing and Communications & IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC)*, Nov 2013, pp. 1246–1253.

[5] M. Besta and T. Hoefler, "Slim Fly: A Cost Effective Low-Diameter Network Topology," in *Proc. of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, Nov 2014, pp. 348–359.

[6] J.-Y. Shin, B. Wong, and E. G. Sirer, "Small-World Datacenters," in *Proc. of the Symposium on Cloud Computing (SoCC)*, Oct 2011, pp. 2:1–2:13.

[7] M. Koibuchi, H. Matsutani, H. Amano, D. F. Hsu, and H. Casanova, "A Case for Random Shortcut Topologies for HPC Interconnects," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, Jun 2012, pp. 177–188.

[8] A. Singla, C.-Y. Hong, L. Popa, and P. B. Godfrey, "Jellyfish: Networking Data Centers Randomly," in *Proc. of the USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, Apr 2012, pp. 225–238.

[9] "Graph golf: The order/degree problem competition," http://research.nii.ac.jp/graphgolf/.

[10] Ü. Y. Ogras and R. Marculescu, ""It's a Small World After All": NoC Performance Optimization Via Long-Range Link Insertion," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol. 14, no. 7, pp. 693–706, Jul 2006.

[11] H. Yang, J. Tripathi, N. E. Jerger, and D. Gibson, "Dodec: Random-Link, Low-Radix On-Chip Networks," in *Proc. of the IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2014, pp. 496–508.

[12] W. J. Dally and C. L. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Transactions on Computers (TC)*, vol. C–36, no. 5, pp. 547–553, May 1987.

[13] D. M. Chiu, M. Kadansky, R. Perlman, J. Reynders, G. Steele, and M. Yuksel, "Deadlock-free Routing Based on Ordered Links," in *Proc. of the Annual IEEE Conference on Local Computer Networks (LCN)*, Nov 2002, pp. 62–71.

[14] M. Ebrahimi and M. Daneshtalab, "EbDa: A New Theory on Design and Verification of Deadlock-free Interconnection Networks," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, Jun 2017, pp. 703–715.

[15] C. J. Glass and L. M. Ni, "The Turn Model for Adaptive Routing," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, May 1992, pp. 278–287.

[16] W. Qiao and L. M. Ni, "Adaptive Routing in Irregular Networks Using Cut-Through Switches," in *Proc. of the International Conference on Parallel Processing (ICPP)*, Aug 1996, pp. 52–60.

[17] M. D. Schroeder, A. D. Birrell, M. Burrows, H. Murray, R. M. Needham, and T. L. Rodeheffer, "Autonet: A High-speed, Self-configuring Local Area Network Using Point-to-point Links," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 8, pp. 1318–1335, Oct 1991.

[18] F. Silla and J. Duato, "Improving the Efficiency of Adaptive Routing in Networks with Irregular Topology," in *Proc. of the International Conference on High-Performance Computing (HiPC)*, Dec 1997, pp. 330–335.

[19] T. Skeie, O. Lysne, and I. Theiss, "Layered Shortest Path (LASH) Routing in Irregular System Area Networks," in *Proc. of the IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Apr 2002, pp. 194–201.

[20] T. Skeie, O. Lysne, J. Flich, P. Lopez, A. Robles, and J. Duato, "LASH-TOR: A Generic Transition-Oriented Routing Algorithm," in *Proc. of the International Conference on Parallel and Distributed Systems (ICPADS)*, Jul 2004, pp. 595–604.

[21] G.-M. Chiu, "The Odd-Even Turn Model for Adaptive Routing," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 11, no. 7, pp. 729–738, Jul 2000.

[22] A. Jouraku, M. Koibuchi, and H. Amano, "An Effective Design of Deadlock-Free Routing Algorithms Based on 2D Turn Model for Irregular Networks," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 18, no. 3, pp. 320–333, Mar 2007.

[23] Eric Weisstein, "Topological Sort." MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/TopologicalSort.html.

[24] N. Jiang, D. U. Becker, G. Michelogiannakis, J. Balfour, B. Towles, J. Kim, and W. J. Dally, "A Detailed and Flexible Cycle-Accurate Network-on-Chip Simulator," in *Proc. of the IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, Apr 2013, pp. 86–96.

[25] H. Matsutani, P. Bogdan, R. Marculescu, Y. Take, D. Sasaki, H. Zhang, M. Koibuchi, T. Kuroda, and H. Amano, "A Case for Wireless 3D NoCs for CMPs," in *Proc. of the Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 2013, pp. 22–28.

[26] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004.