Fault-Aware Load-Balancing Routing for 2D-Mesh and Torus On-Chip Network Topologies

Pengju Ren, *Member, IEEE*, Michel A. Kinsy, *Member, IEEE*, and Nanning Zheng, *Fellow, IEEE*

Abstract—Routing algorithm design for on-chip networks (OCNs) has become increasingly challenging due to high levels of integration and complexity of modern systems-on-chip (SoCs). The inherent unreliability of components, embedded oversized IP blocks, and fine-grained voltage-frequency islands (VFIs) management among others, raise several challenges in OCNs: (a) network topologies become irregular or asymmetric making circular route dependencies that lead to deadlock hard to detect; and (b) routing algorithms that lack strong load-balancing properties often saturate prematurely. In order to address the aforementioned deadlock and load-balancing problems, we propose the traffic balancing oblivious routing (TBOR) algorithm. It is a two-phase routing algorithm consisting of: (1) construction of the weighted acyclic channel dependency graph (CDG) for the OCN to efficiently maximize available resource utilization; and (2) channel ordering across turn models to keep the underlying CDG cycle-free to guarantee deadlock-freedom using one or more turn-models. Channel bandwidth utilization and traffic balancing are achieved through static virtual channel allocation according to residual bandwidth of healthy links. In addition, we introduce in this work two schemes of different granularity of fault detection and analysis while guaranteeing in-order packet delivery by assigning a unique path to each flow. Extensive experiments demonstrate the proposed routing methodology outperforms previous algorithms.

Index Terms—Fault-tolerant routing, irregular network, load balance, resource utilization, channel dependency graph

1 Introduction

As technology scales down to the nanoscale regime, the integration of billions of transistors into a single chip has become commonplace. Computer designers are turning to multicore and many-core systems, which not only have the parallel computing benefits, but also provide functional redundancy, that enables compute resource management and reconfigurability to address the issues of reliability and fault tolerance [1], [2]. Architectures with several distinct CPU cores on a single die are now standard: general-purpose processors can include as many as fifteen cores [3] and multicore designs with 64 or more cores are commercially available [4].

Due to the lack of scalability in bus-based protocols, on-chip networks (OCNs) have been introduced as an effective data communication infrastructure for systems-on-chip (SoCs) and multicore or many-core systems [5]. For many applications, irregular on-chip-network topologies are required in order to integrate different sizes of IP blocks from different vendors [6], [7]. These irregular networks play an important role in ensuring the reliability, availability, low-latency and high-throughput requirements. In multicore systems built at 65 nm and below, it is extremely difficult to move signals across the die in a single clock cycle. As a consequence, globally asynchronous locally synchronous (GALS) schemes have been introduced for better power efficiency and design modularity [8], [9]. And voltage-frequency islands (VFIs) are used to perform fine-grained power management [10], [11]. In these designs, the system is partitioned into multiple VFIs. Each VFI works on different supply and threshold voltage levels. The VFI ON/OFF power mode is used to optimize the chip power consumption based on computing workload variations during the application execution.

In addition, the unreliability of components has emerged as one of the fundamental barriers to future scaling. There are two types of component variations: static (or permanent) and dynamic (or transient). The former are caused by random dopant fluctuations and sub-wavelength lithography during fabrication, while the latter are caused by particle strikes, voltage and temperature variations, crosstalk and electromagnetic interference during runtime [12], [13].

Unreliability of components, embedded oversized IP blocks (OIPs) and fine-grained ON/OFF power management of VFIs present several key challenges for the design and implementation of OCNs, as shown in Fig. 1. Some of the routers and/or links may not be available due to unpredictable fault distributions, different dimensions of IP blocks and dynamic power management of VFIs or a combination thereof. To avoid packet loss or deadlock scenarios that arise due to topology changes caused link or router failures, OIPs, or VFIs, it's vital for the OCN to possess resilient communication and load-balancing properties. Moreover, in the event of partial available communication

- *P. Ren and N. Zheng are with the Xi'an Jiaotong University, Xi'an, Shaanxi 710049, P.R., China. E-mail: pengjuren@gmail.com, nnzheng@mail.xjtu.edu.cn.*
- *M.A. Kinsy is with the Department of Computer and Information, University of Oregon, Eugene, OR 97403. E-mail: mkinsy@cs.uoregon.edu.*

Manuscript received 20 June 2014; revised 16 May 2015; accepted 20 May 2015. Date of publication 31 May 2015; date of current version 10 Feb. 2016. Recommended for acceptance by H. Sarbazi-Azad.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2015.2439276

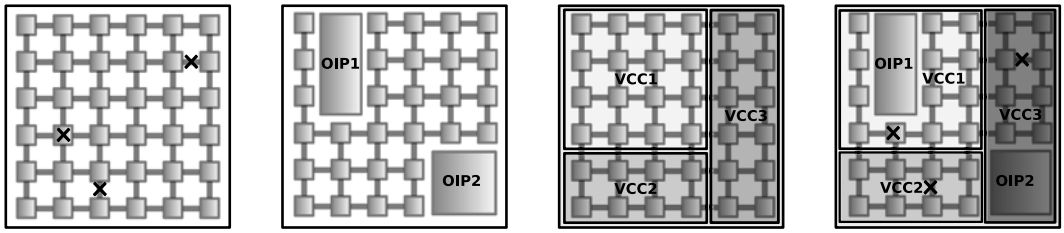


Fig. 1. (a) A faulty 6×6 2D-mesh with one node and two links are broken; (b) An irregular mesh with two oversize IP blocks: OIP1 and OIP2; (c) A network with three voltage-frequency islands: VCC1, VCC2 and VCC3, each of them can be powered ON/OFF independently; (d) A hybrid example of all three cases.

resources, the on-chip network should retain the availability of as many computing resources as possible and ensure graceful performance degradation. In particular, it must be (1) flexible enough to allow a variety of faulty and fault-free configurations [14], (2) capable to provide the required adaptivity for integrating different sized IP blocks, and (3) suitable for different fine-grained power management strategies on VFIs.

Fault-tolerant routing algorithms are the most practicable and commonly proposed solutions, because of the similarity between faulty and irregular on-chip networks [15], [16]. The general approach in this domain is to route packets through available components or resources and to avoid out-of-service links and routers (or OIPs and power-off VFI blocks) using adaptive or table-based oblivious routing schemes. Although it is worth noting that our proposed methodology will also apply to networks with OIPs and power-off VFI blocks. Most previous works have mainly focused on fault avoidance but often overlooked load-balancing and power distribution issues. In these works, fault-tolerant routers are either fully operational or out-of-service even when only a single virtual channel (VC) or switch of the crossbar is defective. These approaches are highly inefficient, wasteful and costly [17]. Furthermore, traffic load imbalance may cause hotspots in the network, and according to Aisopos et al. [18], high temperatures result in 4–10 percent fault probabilities (≥100℃ up to 125℃) which further exacerbates the reliability problem. A good alternative is to operate semi-faulty routers or links in partial-usage modes by making use of remaining available network resources, such as healthy VCs and crossbar switches and to re-route packets around faulty components. Our results show that this approach provides better resources utilization and network performance even under severe fault scenarios.

The oblivious fault-tolerant algorithm proposed in this work is for wormhole routing. Deadlock avoidance is guaranteed by combining appropriate turn models for single or multiple virtual set(s) with priority. The virtual channel allocation is done explicitly through the routing function. Traffic load-balancing is based on the global knowledge of the underlying irregular network topology and the communication characteristics of applications [19], [20]. Resource utilization adjustments for faulty networks are done through specific fine-grained fault information analysis. Route selections and router configurations are performed statically. They occur when (1) a new permanent fault is detected and it must be reflected in a modified topology, (2) a new ON/OFF power management strategy is introduced, or (3) a new application is starting. The proposed routing methodology achieves this without introducing significant additional penalty or cost in terms of extra latency and hardware resource (VC number), while preserving the in-order delivery. Additionally, we are developing algorithms to identify the minimum number of VCs to satisfy communication requirements of an application given the topology of the irregular network.

In the remainder of the manuscript, we provide preliminaries and definitions in Section 2. In Section 3, we outline our methodology and describe the TBOR routing algorithm and implementation. Related works are presented in Section 4. Section 5 evaluates TBOR performance and compares it with other routing algorithms. Section 6 concludes the paper.

2 System Framework

2.1 Router Micro-Architecture

Our discussion assumes a state-of-art credit-based virtual-channel router on an irregular 2D-mesh or torus network. Each input port has several buffers, where packets are stored until the route computation and VC allocator units determine the next hop and associated VC. The switch allocator selects which flits will traverse the crossbar. Previous works assume the router is fully unavailable as long as one of its components is identified as faulty. This approach of labeling the whole router as out-of-service is inefficient since operational resources are also disregarded in the process. In our routing framework, we develop two techniques to handle different fault granularities. One is a coarse-grained technique, which, like previous works, treats the router fully out-of-service if one of its components is broken. The other is a fine-grained method, which treats the router as semi-faulty and uses the remaining functionality of the partly defective router to route application flows.

2.2 Irregular Network Model

A network topology can be characterized by a set of nodes connected by links or channels. For a K by K 2D mesh, node N is indexed by the tuple (N_x, N_y), where $0 \leq N_x < K-1$ and $0 \leq N_y < K-1$. We denote the directional link from node A to B by $[A, B]$. A unidirectional channel associated with each physical link from node A to node B is represented by $\langle A, B\rangle^i$, where i indicates which VC set it belongs to. Therefore, a path connecting nodes S and D can be represented with an ordered set of channels.

Fault control involves two basic steps: fault detection (or diagnosis) and fault tolerance (or containment). A dedicated built-in self-test (BIST) mechanism could pinpoint faulty links or components in routers. It is done by allowing a node to send probe signals to its neighbors and to mark neighboring routers or router components defective if acknowledgment is not received within a certain time interval. Fault diagnosis is an orthogonal topic to fault containment and will not be addressed in this paper.

Faults inside the router can affect the routing table or the switch allocator or both. These faults can lead to misrouting and starvation. Corrupted packets are detected via cyclic redundancy check (CRC) and are discarded. When an error persists over several cycles, the component is identified as permanently faulty and labeled out-of-service. We consider permanent faults that occur both at the router and link levels. When a physical link fails, all associated VCs are marked faulty, that is: if link $[N, M]$ is faulty, channels $\langle N, M\rangle^j$ where $j = 0, 1 \ldots I-1$ are all treated as out-of-service (I is the number of VC sets). For our coarse-grained scheme, all physical links connected to a partially faulty router are made unavailable, therefore rendering the router fully out-of-service. With our fine-grained approach, a router with defective components is still used at reduced capacity to perform limited functions, details are presented in Section 3.2.1. The router is considered completely broken if all of its links are faulty.

2.3 Turn Models and Degree of Adaptiveness

Deadlocks occur when messages or packets waiting to be delivered form a cycle. Glass and Ni [21] proposed a well-known solution called the turn models that places restrictions on routing path turns as a mean to break link dependencies and to guarantee cycle-free adaptivity. Chiu presented odd-even routing [22], an extension of the turn models that does not allow turn pairs of East-North and North-West or East-South and South-West in same column, thus prohibiting circular paths.

Recently, Shafiee [23] and Fu [24] proposed similar protocols called Extended Turn Model (ETM) and Abacus Turn Model (AbTM) respectively. In the AbTM routing scheme, there are two nodes specified as *clockwise bead* and *counter-clockwise bead*. They selected for each column except the left-most column. In this scheme, East-South and South-West turns are forbidden for all nodes above and below *clockwise bead*; and North-West and East-North turns are prohibited for nodes above and below *counter-clockwise bead*.

Turn model based routing schemes are partially adaptive routing algorithms, which means the number of shortest paths from source to destination is determined by which turn model is taken and locations of the source and destination. As Glass and Chiu describe in [21] and [22] the degree of adaptiveness of a turn model routing algorithm is as follows:

$$S_{West-First} = \begin{cases} \frac{(\Delta x+\Delta y)!}{\Delta x!\Delta y!} & \text{if } D_x \geq S_x \\ 1 & \text{otherwise,} \end{cases}$$

$$S_{north-last} = \begin{cases} \frac{(\Delta x+\Delta y)!}{\Delta x!\Delta y!} & \text{if } D_y \leq S_y \\ 1 & \text{otherwise,} \end{cases}$$

$$S_{negative-first} = \begin{cases} \frac{(\Delta x+\Delta y)!}{\Delta x!\Delta y!} & \text{if } (D_x - S_x)(D_y - S_y) \geq 0 \\ 1 & \text{otherwise,} \end{cases}$$

$$S_{odd-even} = \frac{(\Delta y+h)!}{\Delta y!h!} \quad h = \left\lceil \frac{\Delta x}{2} \right\rceil or \left\lceil \frac{\Delta x-1}{2} \right\rceil,$$

where $S_{algorithm}$ represents the number of the shortest paths (minimal routes) allowed from source (S_x, S_y) to destination (D_x, D_y), $\Delta x = |D_x - S_x|$ and $\Delta y = |D_y - S_y|$. With partial routing adaptivity, turn model based routing can potentially achieve fault tolerance by providing alternative choices when messages encounter faulty regions. However, with a given turn model, the degree of adaptivity (path diversity) is completely determined by the locations of the source and destination pairs. Similarly, with ETM and AbTM routing, the $S_{algorithm}$ depends on the position of the selected *clockwise* and *counter-clockwise bead*, and the position of faulty links may disconnect some node pairs. We will highlight some limitations associated with using turn model based adaptive routing for fault tolerance in Section 3.2.2.

2.4 Definitions

For a given application and an underlying irregular network, the objective of the work being proposed is to decide how packets should be routed without deadlock, while maximizing the available communication resources and balancing traffic to attain good performance. Within this framework, we will make use of the following definitions:

Definition 1. *Given an OCN architecture, there is an associated directed graph (ARCG) $G = G(R, Ch)$, where R and Ch represent the set of routers and channels in the network. Each $r_i \in R$ is a router and is connected to a processing element (PE). Each directed arc $Ch_{i,j} \in Ch$ represents a set of VCs from r_i to r_j, $\forall ch_{i,j} \in Ch_{i,j}$, $C(ch_{i,j})$ gives the bandwidth available on the channel.*

For a given target application running on a fixed OCN architecture, several problems must be addressed: (a) which processing element should each application module be mapped to and (b) how should packets be routed to achieve the best communication performance while using minimal energy and meeting other constraints like area overhead, temperature, etc. However, the problem of finding an optimal assignment of modules to processing elements has been shown to be NP-hard [25], [26]. The focus of the project will be on fault-aware routing in irregular networks and we will assume that application modules have already been bound to the processing elements.

Definition 2. *A channel dependence graph (CDG), denoted by $CDG(V, E)$, is derived from the ARCG, where a vertex $v_{i,j} \in V$ corresponds to an edge $ch_{i,j}$ in ARCG. There is a directed arc from $v_{i,x}$ to $v_{x,j}$ if $ch_{i,x}$ and $ch_{x,j}$ are input and output channels of the same node x (ignoring 180 degree turns)—in other words, if a packet can be routed from $ch_{i,x}$ to $ch_{x,j}$ without traversing any other edges in the ARCG.*

In a faulty network that employs a fine-grained fault-diagnosis scheme, the vertices of CDG are the available communication channels, therefore, they do not include any faulty channels and there is no arc between two vertices (channels) if the switch link of the crossbar connecting them

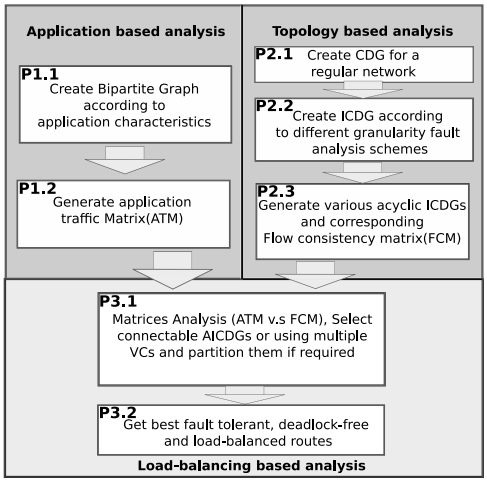


Fig. 2. Flow of the TBOR routing algorithm: the generated channel dependence graph of the irregular network (ICDG) is different for the fine- and coarse-grained techniques.

is disabled. The specific procedure for constructing CDGs from defective networks is discussed below.

3 CDG Based Exploration and TBOR Routing Algorithm

The outline of our routing approach and the TBOR algorithm are shown in Fig. 2:

Phase1: Application based analysis

- **P1.1:** Application-based analysis results in a bipartite graph.
- **P1.2:** Generate the corresponding application traffic matrix (ATM), which represents all the communication requirements of a given application and is independent of the underlying network topology.

Phase2: Topology based analysis

- **P2.1, P2.2:** A network topology analysis generates fine- or coarse-grained faulty weighted acyclic channel dependency graphs (AICDGs), derived from the irregular network channel dependency graph (ICDG) by potentially deleting some edges to remove cyclic dependencies.
- **P2.3:** Create the flow consistency matrix (FCM). A FCM contains all of the connectivity information of the underlying acyclic network topology independent of any application that runs on it.

Phase3: Load-balancing based analysis

- **P3.1:** Then, according to the application's communication properties (ATM and bandwidth demands) and optional network connectivity properties (FCMs), we select appropriate turn model(s) for the VC(s), using one or more VC set(s) and partitioning them by priority to meet the application communication requirements.
- **P3.2:** Finally, we use Dijkstra's weighted shortest-path algorithm to determine the routes for each flow, and the best load-balanced routes are obtained by comparing channel utilization and minimal maximum channel loads among all the results. Then, we use table-based routing and static VC allocation to assign communication resources to each flow.

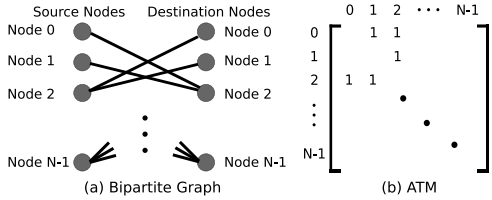


Fig. 3. Application traffic matrix derived from Bipartite graph, represent all the communication behavior of application.

The connectivity of the irregular network is reflected in the underlying ICDG—in other words, node pairs can communicate with each other as long as there is at least one path connecting them in the ICDG. Otherwise, transmission between them is impossible. We are exploring different turn models by deleting edges from the CDG to construct acyclic turn model based AICDGs. There are several ways of removing cycles and constructing acyclic CDGs. Our methodology is not confined to turn models. It is amenable to other cycle-breaking techniques even *ad hoc* or random.

3.1 Application Communication Analysis

A bipartite graph can be used to represent the communication behavior of a particular application [27]. Suppose the number of nodes in the network is N, with nodes on the opposite sides of the bipartite graph representing message sources and destinations respectively. Edges between sources and destinations indicate communication requirements, as shown in Fig. 3a. Therefore, the bipartite graph contains all of the communication information of the given application and it is independent of the underlying network topology. From the bipartite graph we will generate the corresponding application traffic matrix (Fig. 3b).

3.2 Network Topology Based Analysis

3.2.1 Acyclic ICDG without Virtual Channel

Out-of-service links and routers can cause an network topology to become irregular and asymmetric, and can make routing algorithms much more complex compared to fault-free networks. The randomness and unpredictability of fault distribution also makes it hard to detect circular dependencies of routes. The formulation of the deadlock properties of our routing algorithm is inspired by Dally and Seitz [28].

Fig. 4 shows CDGs for a 3×3 2D mesh without virtual channel: (a) faulty mesh, defective router is 3 (south input buffer and North to East switch link of the crossbar are broken); (b) and (c) are ICDGs generated by coarse- and fine-grained schemes respectively; (d) and (e) are relevant West-First turn model based AICDGs using coarse- and fine-grained techniques; (f) and (g) are AbTM based AICDGs using coarse- and fine-grained diagnosis methods where the selected *clockwise bead* is 1 and 8, *counter-clockwise bead* is node 2 and 4. All channels are directional as indicated by the arrows (without 0- and 180-degree turns), notice the more available communication resources using fine-grained fault diagnosis are highlighted with blue circles and dotted lines; (h), (i), (j) and (k) are corresponding flow consistency

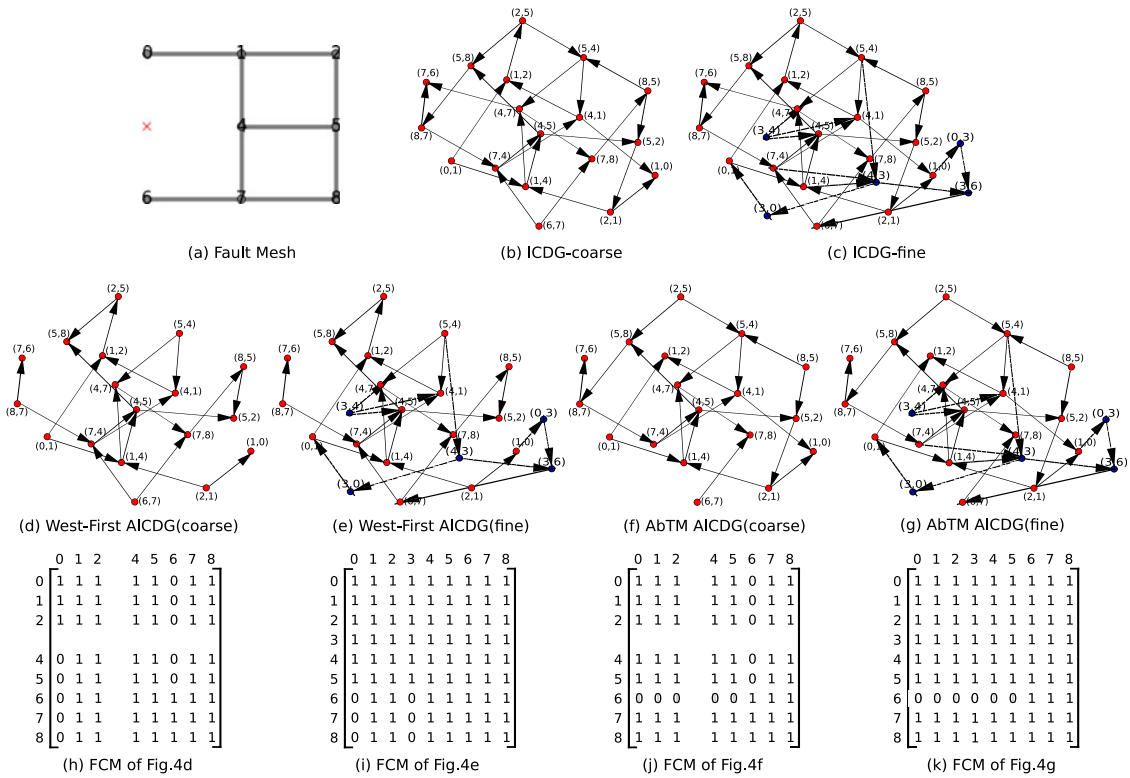


Fig. 4. CDGs for a 3×3 2D mesh without virtual channel.

matrices, element "1" means there exists at least one path that could connect related nodes, and we are assuming node could talk to itself for brevity. In (h) and (j) there is no row and column elements associated with router 3, because for the coarse-grained scheme, router 3 is treated as fully out-of-service.

In the coarse-grained scheme, we regard the defective router as fully out-of-service. If node 3 is faulty, then links $(0,3),(3,0),(3,4),(4,3),(3,6),(6,3)$ and associated edges are deleted from the CDG, as shown Fig. 4b. For the fine-grained approach, all the healthy buffers and available crossbar switches of the defective router are kept, and only the broken components are removed. Consequently, vertex (6,3) (the broken input buffer from south node 6 at router 3) and the edge between (0,3) and (3,4) (north to east switch link of the crossbar at router 3) are removed from CDG, Fig. 4c illustrates the ICDG. We remove circular dependencies by deleting edges from the ICDG to generate the acyclic channel dependency graph. Since the unpredictable arrangements of failed components break the symmetry property of a mesh network [2], we are investigating different types of generalized turn models [21], as well as AbTM [24] and ETM [23].

Turn model based adaptive routing removes certain channel dependencies in order to prevent deadlock. It also sacrifices routing path diversity for some source-destination pairs. For irregular network topologies, it can severely affect routing options and limit path diversity. This can cause some flows to not be routed even with non-minimal routing. Limitations of the turn model for routing are reflected in the AICDGs. Fig. 4d shows the West-First turn model based AICDG. There is no path that can connect node 4 to nodes 0 and 6. The same results are reflected in the flow consistency matrix, which represents the connectivity of the AICDG. Labels of the column and row in the FCM represent sources and destinations, the value "1" in cell (i,j) means that source node i can communicate with destination j; otherwise, there is no path from node i to j.

For West-First, Fig. 4d, cell $(4,0)$th is "0", but in the AbTM case, Fig. 4f, messages can be routed from 4 to 0. All the features of connectable node pairs in the AICDGs are demonstrated in the corresponding FCM. Fine-grained analysis offers more routing options than coarse-grained. For example, node 3 is available as the source and destination in 4e and 4g but will be treated as out-of-service in 4d and 4f. Compared to Figs. 4d and 4e, the fine-grained FCM further benefits from partially-operational routers acting as intermediate nodes to provide more available routing options. For instance, marking node 3 partially functional allows communication from node 4 to node 0 and 6 through node 3.

3.2.2 *Restriction of Turn Model without Virtual Channel for Fault-Tolerance*

Intuitively, a network's ability to handle faults can be measured by the number of paths between each source-destination pair allowed by the routing function [14]. In West-First turn model based adaptive routing, if the destination is on the east side of the source ($D_x \leq S_x$), routing will be fully adaptive; otherwise, the degree of adaptiveness is only one. In other words, there will be only one path connecting S with D. Suppose there is a defective row link sitting between node (D_x, S_y) and (S_x, S_y), it is not possible to route from node S to D using the West-First based adaptive routing, even though they are in fact physically connected. Similar restrictions occur for other turn models (cf. Table 1).

TABLE 1
Unavailable Link Positions Causing Unroutable Flows for the Turn Model Based Adaptive Routing

Turn model[1]	affected flow (Source and Destination Relationship)	Unavailable link position
West-First	$S_x > D_x$	$[(L_x, S_y), (L_x + 1, S_y)]$, where $D_x \leq L_x < S_x$
North-Last	$S_y > D_y$	$[(D_x, L_y), (D_x, L_y + 1)]$, where $D_y \leq L_y < S_y$
Negative First	$S_x < D_x$ or $S_y > D_y$	$[(D_x - 1, D_y), (D_x, D_y)]$ and $[(D_x, D_y), (D_x, D_y + 1)]$
	$D_x < S_x$ or $D_y > S_y$	$[(S_x - 1, S_y), (S_x, S_y)]$ and $[(S_x, S_y), (S_x, S_y + 1)]$
ODD-EVEN	$S_x > D_x$ and $S_x\%2! = 0$	$[(S_x - 1, S_y), (S_x, S_y)]$
	$D_x > S_x$ and $D_x\%2 == 0$	$[(D_x - 1, D_y), (D_x, D_y)]$

[1] *In AbTM and ETM, routing restrictions depend both on the position of source, destination and the selected clockwise and counter-clockwise* bead.

3.3 Methodology for Making Irregular Networks Accommodate Application Requirements

3.3.1 ATM versus FCMs

In some circumstances, an AICDG can meet all the communication requirements for the application even without virtual channels at the links. Specifically, if all the "1" elements in the ATM can find a matching "1" in the same position in the corresponding FCM of the AICDG, a network without virtual channels will be sufficient for transmission. We describe the procedure for selecting the qualified FCMs (AICDGs) without virtual channel in Algorithm 1. In practice, the communication requirements of specified applications running on customized platforms is seldom all-to-all (ATM is an all 1's matrix). For example, in multi-processor SoC (MPSoC) most applications [29], [20] have predefined traffic flows, which gives a fair amount of flexibility in choosing qualified AICDGs. For applications without a priori knowledge, it is sufficient to assume that the ATM is an all 1's matrix, meaning that all nodes need to talk to each other.

Algorithm 1. Get Qualified FCM (AICDG) Meet All the Communication Requirements, without Using VCs

Input: Application traffic requirement and defective mesh
Output: Set contains all the qualified $FCMs$
Create ATM from particular application
Generate different turn model based AICDGs, and the corresponding FCM^k, $k = 0 \ldots 19$
SET_{FCM} *contains all the FCMs*
foreach $T_{i,j} == 1$ in ATM **do**
  foreach FCM^k in SET_{FCM} **do**
    if $a^k_{i,j}! = 1$ **then** /* $a^k_{i,j}$ `is element in` FCM^k */
      remove FCM^k from SET_{FCM};
if SET_{FCM} *is not empty* **then**
  return SET_{FCM}
else
  return Null

However, given a network in a severe fault scenario, the FCM could be very sparse, so there is a strong possibility that a single FCM cannot meet all the traffic requirements. In order to mitigate these situations and to provide better network performance, we are investigating and proposing algorithms that use multiple VC sets and apply suitable cycle-break strategies to each set. Simply conjoining different VC sets, however, could result in deadlock among them. Fig. 4a shows an example path: $\{\langle 2,1\rangle^0, \langle 1,4\rangle^0, \langle 4,5\rangle^1, \langle 5,2\rangle^1, \langle 2,1\rangle^0\}$ where a message is first transmitted within VC0, then it is switched from VC0 to VC1 at node 4, and finally back to VC0 at node 2, and creating a cycle in the process. A detailed deadlock-free approach using multiple VCs is presented in Section 3.3.2.

3.3.2 Acyclic ICDG with Multiple Virtual Channels

The procedures of generating the AICDG with multiple VCs and the proof of deadlock avoidance are as follows:

1) Partition virtual channels into sets by priority, so that the v channels associated with the same physical link are divided into v distinct sets ordered by priority.
2) Apply a proper turn model for each virtual channel set to prevent deadlock happening within the set (180-degree turns are ignored). Each separate set would generate a corresponding AICDG.
3) Transitions are only allowed from high to low priority to avoid the possibility of cycles between VC sets. The separate AICDGs for different sets are coupled by adding edges that transmit from higher to lower priority sets.
4) Sets with the same turn model and adjacent priority can traverse each other using either 0-degree turns or appropriate turns. In this particular scenario, we need to add the relevant edges that transmit from lower to higher priority. Suppose, for example, that we are using four VC sets. Applied turn models and priorities of the various VC sets are: West-First (VC0) > East-Last (VC1) > East-Last (VC2) > North-Last (VC3). Since VC1 and VC2 use the same turn model and are with adjacent priority, they can communicate using straight paths (0-degree) and all the turns allowed in East-Last.

Lemma 1. *Multiple virtual channels AICDG is deadlock free.*

Proof. According to Dally and Seitz [28], a routing algorithm is deadlock-free if the channels can be numbered and every packet traverses channels in a strictly descending (or ascending) order. Without loss of generality and for simplicity, we consider the case of two VCs, each applying a different turn model. Virtual channel set VC0 has the higher priority and obeys the West-First turn model, and set VC1 uses the West-Last turn model. We number virtual channels starting at node (x, y) in VC0 using:

$$\begin{array}{ll} \text{North:} & (K-1)(2K-2-2x)+K-2-y \\ \text{South:} & (K-1)(2K-2-2x)+y-1 \\ \text{West:} & (K-1)(2K-2+x) \\ \text{East:} & (K-1)(2K-3-2x). \end{array}$$

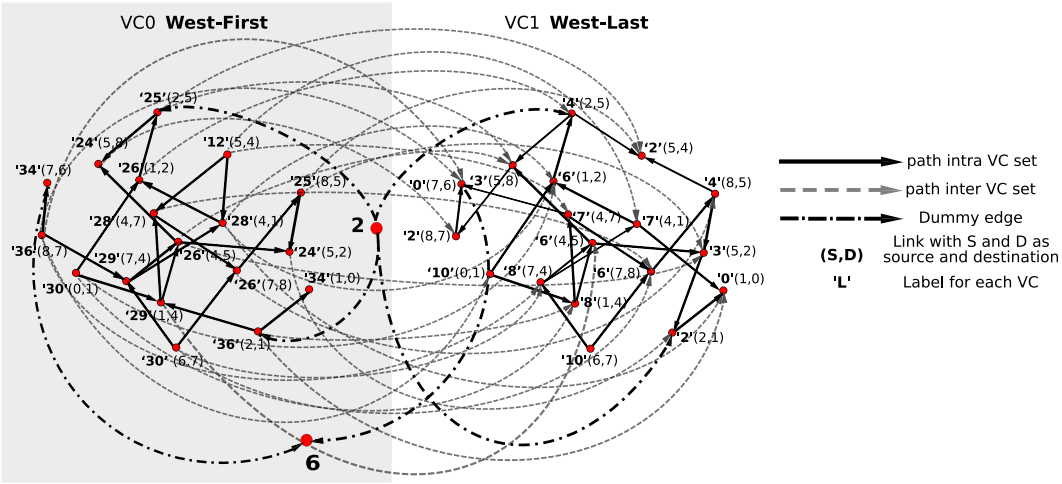


Fig. 5. An AICDG constructed with two VC sets: VC0 and VC1 apply West-First and West-Last respectively. VC0 is assigned a higher priority than V1. The VC labels are in bold type. Dummy nodes (2 and 6) and edges are added and set to dotted when making a route from node 2 to 6.

For the West-Last model, we assign virtual channels in VC1 using:

$$\begin{array}{ll} \text{North:} & (K-1)(3K-2x-4)+y \\ \text{South:} & (K-1)(3k-2x-3)-y \\ \text{West:} & (K-1)(3k-4-2x) \\ \text{East:} & (K-1)(x-1). \end{array}$$

□

This way, packets routed within the same VC set are guaranteed to be in strict descending order. We add $L = 4K(K-1)$ to the channel labels in set VC0, where L is equal to the number of unidirectional physical links. It is also equal to the number of VCs in a set for a K by K 2D mesh. In general for each set VCi, we add $(I-i-1)L$ to the channel labels belonging to the set, where I is the number of VC sets. This approach guarantees that the highest label in a lower priority set is smaller than the lowest label in any higher set. Consequently, even packets routed across different VC sets still travel in strictly descending label order, therefore providing deadlock avoidance. For other turn models, we can rotate this labeling paradigm clockwise or counter-clockwise, or reverse channel direction to obtain the corresponding labeling. We connect them using the same strategy as in West-First and West-Last. Fig. 5 shows an AICDG with two VC sets. The irregular network is shown in Fig. 4a. The AICDG is constructed using the approach described above. In this example, K is 3 and we add 24 to the labels of channels belonging to VC0. All channels are directional with arrows (without 180 degree turns); dashed arrows indicate allowed paths from VC0 to VC1.

3.3.3 Connecting Multiple FCMs to Meet the Communication Requirements of the ATM

As long as all the node pairs are physically connected in the underlying ICDG, we apply proper turn models to each VC set and couple them together following the methodology described above. In this way, we can overcome the restriction of turn models without VCs. Connecting as many flows as possible, and adopting more VC sets if desirable, all the communication flows can be connected to each other while preserving freedom from deadlock. Finding complementary VC partitions can be derived from FCMs. Consider routing messages from node i to j, suppose $(i,j)\text{th} == 0$ in all the FCMs, this means that a single turn model cannot connect node i with j. However, if there exist a node u, where $(i,u) == 1$ in the first FCM(VC0) and $(u,j) == 1$ in the second FCM(VC1), messages are routed from node i to u using set VC0, then at node u they are switched to set VC1, and traversed from u to j. A detailed procedure for finding the complement FCMs for universal application (ATM is all 1's matrix) is described in Algorithm 2. Applications with refined traffic requirements just need a slightly modification on Algorithm 2. Furthermore, Algorithm 1 and 2 can also be used to determine the minimum number of VC sets to meet application communication requirements.

Algorithm 2. Finding the Complement FCM Pairs for "all-to-all" Communication Requirement

Input: Irregular Network
Output: Complement FCM pairs for "all-to-all" communication requirement
Generate different turn model based AICDGs with associated FCM^k;
SET_{FCM} *contains all the* $FCM^k s$;
Loop1: **foreach** FCM^k in SET_{FCM} **do**
  copy $SET^k_{FCM} = SET_{FCM}$;
  remove FCM^k *from* SET^k_{FCM}
  Initialize SET^k *as an empty set*
  foreach $a^k_{i,j}$ in FCM^k: **do**
    if $a^k_{i,j} == 0$ **then**
      if $a^k_{i,n} == 0\ (n = 0 \ldots K)$ **then**
        go to **Loop1**
      else
        SET^k *contains all the* $a^k_{i,n} == 1$
foreach $a^k_{i,n}$ in SET^k **do**
  loop2: **foreach** FCM^l in SET^k_{FCM} **do**
    if $a^l_{n,j} == 0$ **then**
      remove FCM^l from SET^k_{FCM} *go to* ***loop2***
if SET^k_{FCM} is not empty **then**
  Return SET^k_{FCM}
else
  Return Null

For example, although routing from node 2 to 6 is not feasible (Fig. 4d). We can, however, first route from 2 to 8

using VC0 (applied West-First turn model), and then switch to VC1 (applied West-Last turn model) at node 8 to reach destination 6 to accomplish the desired communication. However, it may not be practical to determine the exact number of VCs sufficient for any application, since the irregular network topology under fault conditions is unpredictable and flows could be affected by multiple regions. Moreover, network topology could be complicated when more faulty components are considered. For some extreme cases, the FCM could be very sparse.

3.4 Weighted Shortest-Path Based Flow Control

Load balancing is the measurement of how uniformly resources across the network are being utilized [14]. It is widely known that traffic imbalance causes some links to stay idle while others are over-utilized, resulting in heavy congestion at the overloaded intersections and premature network saturation. Although some algorithms that split flows across multiple paths spatially [30] or temporally [31] could potentially achieve better throughput, this will come at the cost of out-of-order packet delivery. In this paper we only investigate undivided flow problems, and assign each flow with unique path to guarantee packets are delivered in the original transmission order.

Before running the off-line Dijkstra's weighted shortest-path algorithm [32] to choose the "maximum available bandwidth" path for each flow, we make a temporary modification to the underlying AICDG. For example, for a flow i with source S_i and destination D_i, firstly, we add dummy vertices S_i and D_i, then add direction edges from S_i to all vertices that have S_i as the source, and to D_i from all vertices that have D_i as the destination. Fig. 5 shows an example of traffic flow from node 2 to node 6. The weight function is similar to [29] and [33]: each link starts with an initial normalized weight $W = 1$, and once a flow i routes through it, the link weight will be updated by residual capacity which is:

$$W' = \begin{cases} 1 - \sum_i d_i C & C > \sum_i d_i \\ \infty & C \leq \sum_i d_i, \end{cases}$$

where C is the capacity of link and d_i is the bandwidth demand of flow i. Note that flow assignment follows decreasing bandwidth demand, and thus a minimum-weight path is derived using Dijkstra's algorithm. Dummy vertices and connected edges are removed from AICDG afterwards. This way, instead of choosing the shortest path as usual, flows always pick a path with more available bandwidth as the primary option to avoid congestion, allowing traffic workload to distribute evenly across the network.

Applying different types of turn model and different priority partition for VC sets will generate different AICDGs. In other words, different channel topology graphs may result in different qualities of network performance. We can select the best result among all the AICDGs by computing the minimal maximum channel load and the number of channels with the maximum load. It is worth noting that since the Dijkstra's weighted shortest-path based algorithm is constructed off-line, it does not directly resolve runtime faults.

3.5 TBOR Routing Algorithm System Level Implementation

From the system implementation view, in order to efficiently deal with various OCN component failure patterns under different application traffic, run-time rerouting or route reprogramming are required. Therefore we design the TBOR routing algorithm to be used for the fault detection and analysis. Once the communication resource assignment has been determined according to the TBOR algorithm, route reprogramming can be performed using table-based routing [14] and static virtual channel allocation [34]. The coupling of the TBOR algorithm with table-based routing and static virtual channel allocation requires no additional specialized mechanism or hardware.

3.6 Generalization of the TBOR Routing Algorithm

It is worth noting that the TBOR algorithm is not limited to routing under faulty conditions. It covers more general cases where some of the routers or links are unavailable due to various reasons, e.g., irregularity in topology caused by OIP blocks and off-mode VFIs containing one or more routers.

Figs. 1b and 1d illustrate cases where OIP blocks may contain routers and links classified as out-of-service. Here routers and links associated with OIP1 and OIP2 are unavailable. In Figs. 1c and 1d, when VCC1, VCC2 or VCC3 is powered off, all the communication resources belonging to the VFI become inaccessible. In which case the disabled routers and links are treated in the same manner as faulty components in our framework, although the locations of OIPs and VFIs are known in advance [6], [35].

4 Related Work

Chien and Kim's planar adaptive routing algorithm [36] is able to tolerate any number of faults using three VCs but this approach sacrifices a large number of faultless nodes. Glass and Ni [37] proposed *one-fault-tolerant* routing derived from negative-first routing algorithm without VCs with (n-1) fault-tolerant degree for n-dimensional meshes. Wu [15] presented a dimension-order and odd-even turn model based algorithm for 2D meshes, with the restriction that the fault components cannot be on the first and last two columns of the east and west boundaries.

Block based fault models [38], [39] sacrifice system processing capacity because some fault-free nodes are isolated and marked as faulty in order to form rectangle or convex regions. In addition, packets are routed around fault regions, thus leading to a significantly unbalanced link utilization and degraded network performance.

Based on Duato's protocol [40], Gomez et al. [16] presented a multi-phase routing scheme, where adaptive routing is used and deadlock-freedom is guaranteed by using different escape channel for each phase. The two-phase version of the algorithm requires a minimal of three VCs. In the multiple intermediate nodes version, adaptive routing is disabled and misrouting for some paths is adopted to improve the fault tolerance degree. The multi-phase algorithm performs well because it distributes traffic more evenly across the network. However, it still exhibits some limitations due to the fact that the underlying Duato's

protocol may not make full use of the escape channel's bandwidth.

Gopalan [41] proposed a fault-tolerant load balancing routing named *FTLBR*, by selecting primary and backup routes for virtual private networks with QoS guarantees. They claimed that achieving network-wide load balancing in the routes selection is the key to maximizing network performance. Fick [42] presented a highly resilient fault-tolerant routing, applying flag transmission and a routing entry update mechanism to achieve a high reliability rate. But the fraction of permitted turns may be very low under high failure rates. Ebrahimi [43], [44] proposed various minimal path-based fault-tolerant routings to reduce packet latency and congestion around faulty regions. Giuseppe [45] proposed a selection algorithm that can be coupled with adaptive routing algorithms to relieve congestion. A Q-learning method using two-hop fault information to reconfigure routing tables and to avoid faults was introduced in [46].

However, as pointed out by Ren [47], the unpredictable nature of the fault location may lead to a disconnected network. If one of the faulty routers or links happens to be the *cut-vertex(bridge)* of the network, the failure will disconnect all node pairs belonging to disconnected subgraphs. A misrouting-supporting adaptive routing method based on virtual network partitioning [48] for 2D and 3D mesh, called channel overlapping, has been proposed. This method constructs fault blocks inside separate planes and deadlock freedom is guaranteed by allowing and disallowing some turns in order to break potential cyclic dependencies.

Lotfi-Kamran [49] proposed the BARP-A routing algorithm to balance traffic distribution among all the shortest path output ports. With this approach, routing fails when all the shortest paths become faulty. In [50], Palesi and Daneshtalab gave a good overview of existing routing algorithms, including fault-tolerant routing algorithms, for OCNs and next-generation many-core OCN-based SoCs.

Xanthopoulos [51] and Linder [52] applied channel dependency graphs (CDG) to exploring fault-tolerant adaptive routing; Kinsy [29] took advantage of a weighted CDG to avoid deadlock in a fault-free network. These methodologies form the basis for our work, and we expanded them to irregular and faulty networks. To the best of our knowledge, we are the first to construct elaborate weighted fault CDGs though fine-grained fault-diagnosis to find the minimal number of VCs adequate for fault tolerance. Furthermore, we use a heuristic approach for achieving workload balance across the network.

5 Performance Evaluation and Discussion

For the comprehensive comparative study of the proposed traffic load balancing routing algorithm with previous works, we implement both coarse- and fine-grained versions of the algorithm using two (TBOR(coarse)- or TBOR (fine)-VC2) and four (TBOR(coarse)- or TBOR(fine)-VC4) VCs. In addition, we implement the *uDIREC* algorithm presented by Parikh [53], the f_{DATE09} scheme proposed by Fick in [42] and the intermediate-node based fault-tolerant algorithm (*finter*) proposed by Gomez [16]. The Gomez scheme using one and two intermediate nodes (*finter(I)* and *finter(Ix2)*) are part of the study. To limit the overhead associated with VCs, we kept the maximum number of intermediate nodes at two, since each subpath needs a separate escape channel (subpaths need to share at least one adaptive channel). The *finter* algorithm requires five VCs for more than two intermediate nodes. Experiments show that the TBOR algorithm offers better performance even with fewer virtual channels. The fine-grained based TBOR algorithm achieves better performance then the coarse-grained version.

TABLE 2
Network Configuration Summary

Characteristic	Configuration
Topology	8×8 2D irregular network
Routing	TBOR, *finter*, f_{DATE09} *and uDIREC*
Flit size	64-bit
Router pipeline stages	four
Link bandwidth	1 flits/cycle
Per-hop (link) latency	1 cycle
VCs per port	none, 2, 4
VC buffer size	4, 8, 16 flits
Average packet length	8 flits
Transistor type (DSENT)	LVT (Low V_{th})
Clock frequency (DSENT)	1 GHz
Manufacturing Process (DSENT)	45 nm technology
Traffic workload	BIT-COMPLEMENT, TRANSPOSE, SHUFFLE, UNIFORM-RANDOM, H.264 decoder profile
Warmup cycles	10,000
Analyzed cycles	1,200,000

5.1 Simulation Details

In the experiments, we used HORNET, a cycle-accurate many-core simulator [54]. HORNET integrates DSENT [55] and HOTSPOT 5.0 [56] to enable power and thermal analysis. In our experimental set-up, we operate HORNET in the network-only mode. Table 2 summarizes configurations and traffic patterns used for the experiments. For synthetic benchmarks, all flows have the same bandwidth demands; for the H.264 application, the flow bandwidth requirements are derived through profiling.

We simulated a 8×8 2D mesh with 5, 10, 15, 20, 30 and 40 percent of unavailable links, including fault situations and irregular networks because of embedded OIP blocks and VFIs that are powered off. The positions of the unavailable nodes and links were randomly generated. The positions of OIPs and VFIs partition are predefined.

In the emulation of component availability changes and fault cases, we made the assumption that the number of out-of-service nodes and links ratio is around 1:2 similar to Boppana [38]. For example in the 10 percent inaccessible links case, 11 of the 112 links in the network are deemed faulty and five nodes are disabled. Under the fine-grained scheme, virtual channels and switch links of the crossbar have the same probability of failure in a faulty router. We gave the same total amount of buffer resources to all the schemes in experiment. Moreover, in order to get the performance results independent of the relative distribution of the faulty or unavailable links, we performed 20 simulations for each particular fault rate case.

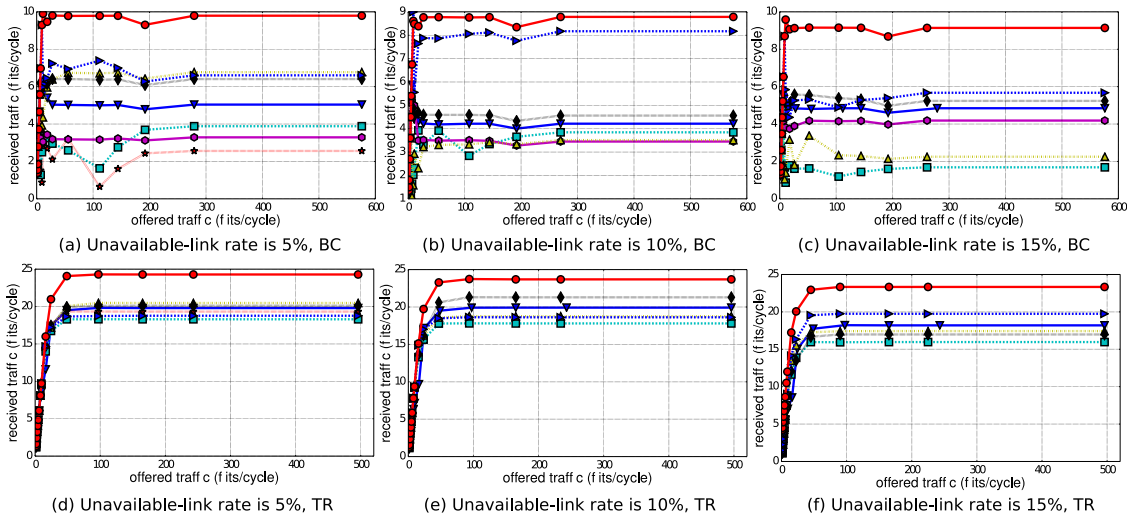


Fig. 6. TBOR(fine)-VC4, TBOR(fine)-VC2, TBOR(coarse)-VC4, TBOR(coarse)-VC2, Finter(Ix2)-VC4, Finter(I)-VC4, f_{DATE09}-VC4. $uDIREC$-VC4. Load-throughput graphs for benchmarks on 8×8 2D mesh with two or four virtual channels. We ran the same experiment on other traffic patterns (SHUFFLE and H.264), the results exhibited the same feature and we omit them here for brevity.[2]

5.2 Traffic Patterns

Synthetic benchmarks and traffic profiles obtained from a parallel implementation of a H.264 decoder are used to investigate network performance. For the coarse-grained scheme, flows with faulty nodes as source or destination are removed. For the H.264 decoder profiles, we re-assigned threads to the available cores. In the fine-grained scheme, a node is unavailable as a source node when all the virtual channels of its input port from the local processing element are broken, or if all the switch links of the crossbar connected with local input port are dead. Similarly, a node cannot be a destination node if all the switch links of the crossbar leading from other input ports to the local output port are out-of-service.

5.3 Throughput Analysis

Fig. 6 represents the average throughput of 20 simulations for each traffic pattern under 5, 10 and 15 percent unavailable-link rates. Note that there are two adaptive channels for *finter* with one intermediate node (*finter(I)*) and only one adaptive channel for the two intermediate nodes case (*finter(Ix2)*).

5.3.1 TBOR versus Others

An intermediate-node based routing algorithm like *finter* always chooses the same qualified intermediate node for different source-destination pairs. Packets belonging to independent flows must compete against each other for the communication resources at intermediate nodes. Heavy congestion at intermediate nodes may lead to performance bottlenecks that will exacerbate when the number of unavailable links increases. Based on our observation, for all the traffic patterns under 10 and 15 percent unavailable-link rates, *finter(I)* can only handle three and two out of 20 simulations. The problem, which is summarized by Gomez in [16], is caused by the inherent limitation of the fault-tolerance degree with only one intermediate node. The f_{DATE09} algorithm determines where turns and links are disallowed based on the network topology to safely route packets around failures, however, it overlooks the load balancing issue.

For the BIT-COMPLEMENT benchmark, the simulation results show an average throughput gain of 97.13 percent for TBOR-VC4 compared to *uDIREC*, Fig. 6. The *uDIREC* algorithm aims at maximizing connectivity of the network by applying Up*/Down* routing. It has relatively larger number of forbidden turns which leads to poor routing flexibility and overall lower network performance. In our experiments, the average percentage of forbidden turns for the *uDIREC* scheme is 20.563 percent for an 8×8 2D-mesh.

In order to maximize the throughput, TBOR can utilize any path—minimal or non-minimal—to transmit a message from its source to destination, and chooses the best solution by comparing the maximum channel loads and number of channels with the maximum load among all the AICDG candidates. The detailed analysis of the network topology, faulty or unavailable links, and algorithms shows that TBOR outperforms all the other algorithms for all the traffic patterns and consistently selects the "maximum available bandwidth" path among all the possible routes. Furthermore, based on the evaluation results, TBOR with only two VCs offers higher throughput than the other algorithms using greater numbers of VCs.

5.3.2 TBOR(Fine) versus TBOR(Coarse)

The results show that in faulty situations the fine-grained scheme always provides more routing paths than the coarse-grained method—Figs. 4d and 4e. The judicious use of partially functional resources allows the TBOR(fine) version of the algorithm to provide better traffic loads distribution and consistently higher throughput when compared to the TBOR(coarse) approach. In the BIT-COMPLEMENT benchmark, we use the *uDIREC* algorithm as the baseline for our

2. The combined turn models of TBOR used for each VC set with priority are: East-Last > North-First > Odd-Even > Negative-First for TBOR(fine)-VC4; West-First > Odd-Even for TBOR(fine)-VC2; Odd-Even > South-First > North-Last > North-First for TBOR(coarse)-VC4; and West-Last > Odd-Even for TBOR(coarse)-VC2.

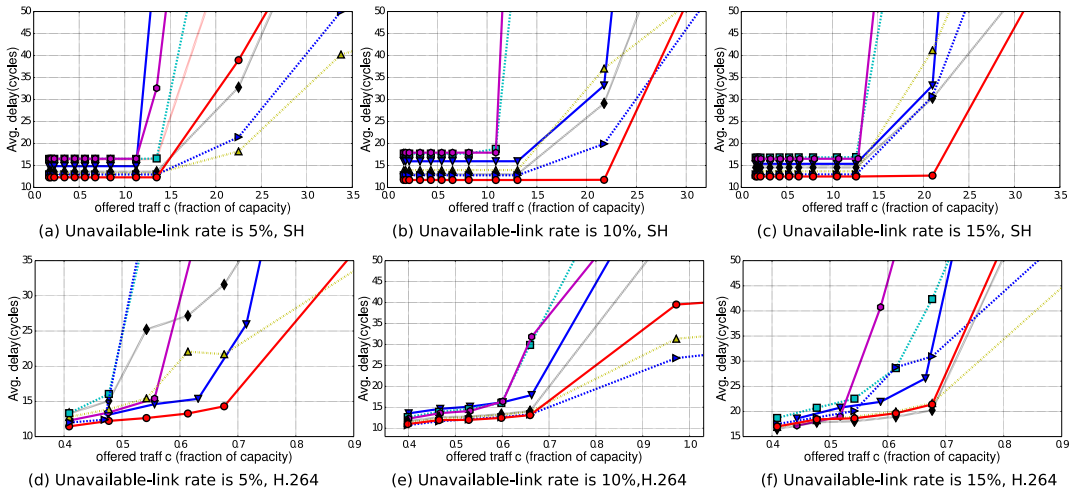


Fig. 7. TBOR(fine)-VC4, TBOR(fine)-VC2, TBOR(coarse)-VC4, TBOR(coarse)-VC2, Finter(Ix2)-VC4, f_{DATE09}-VC4. $uDIREC$-VC4. Latency graphs for benchmarks on 8×8 2D mesh with two or four virtual channels.[3]

discussion because second to TBOR, it achieves the best performance. The coarse-grained TBOR algorithm using four VC has 1.28× throughput with 5 percent unavailable-link rate, 1.1× with 10 percent, and 1.1× with 15 percent unavailable-link rate in comparison to the *uDIREC* scheme. The fine-grained TBOR approach gains over the baseline are 1.96×, 2.09× and 1.89× for 5, 10, and 15 percent unavailable-link rates. The TBOR algorithm is more resilient to faulty or unavailable links and degrades more gracefully with increasing number of inaccessible links when compared with the other algorithms. For the TRANSPOSE benchmark, when the unavailable-link rate is increased from 5 to 15 percent, the *uDIREC* algorithm throughput drops from 19.92 to 18.13 (flits/cycle) – 8.99 percent degradation, the *finter(Ix2)* scheme sinks from 18.53 to 15.86 (flits/cycle) – 14.4 percent degradation, the TBOR(coarse)-VC4 approach falls from 20.38 to 18.86 (flits/cycle) – 7.4 percent degradation and the TBOR (fine)-VC4 throughput decreases from 24.28 to 23.38 (flits/cycle) – only 3.7 percent degradation. These results further highlight the effectiveness of our routing algorithms and the benefits of salvaging partly defective components.

5.4 Latency and In-Order Delivery

Fig. 7 shows the average packet latency for the different benchmarks and routing algorithms. Traffic balancing properties of the TBOR algorithm help prevent some links from reaching premature saturation while large portions of the network remain underutilized. This eliminates some premature uneven local congestions. The average packet latency benefits from better traffic load distribution. In the SHUFFLE benchmark case, the latency of the TBOR-VC4 approach is 20.80 percent lower than the *uDIREC* scheme, 34.95 percent lower than the *finter(Ix2)* algorithm and 32.4 percent lower than the f_{DATE09} technique under unsaturated condition. Notice that the TBOR algorithm is a throughput-driven algorithm which aims at distributing traffic evenly across the network. Routing options are directly proportional to the number of VC sets. In heavily loaded networks, packets may make detours to avoid the congestion. Fig. 7 shows that the TBOR algorithm using four VCs has a higher throughput than the two VCs version but with longer transmission time. For relatively light traffic loads, the TBOR algorithm also produces better average latency than other algorithms, shown in Fig. 8 using the FFT and RADIX SPLASH-2 benchmarks.

In-order packet delivery is a strict requirement for many applications and some higher-level NoC protocols, e.g. cache coherence. The *Finter* scheme with multiple adaptive VCs cannot guarantee in-order delivery, since packets belonging to same flow may be allocated to any available adaptive VCs and may encounter different levels of congestion and network delays. Therefore, there is a high possibility that packets will arrive at the destination node out-of-order. Although reordering of out-of-order packets at the destination can address this problem, it significantly increases latency and resource overheads (reordering buffer and control logic). In-order delivery is ensured by static VC allocation in the TBOR algorithm. Each flow uses a single VC in every phase following a unique path.

5.5 Virtual Channel Restriction and Reliability Analysis

Virtual channels were originally proposed to solve head-of-line blocking and deadlock problems. They are also widely used for fault-tolerant routing [52], [58]. Most virtual channel based routing algorithms for irregular networks impose restrictions on the shape, location, number of unavailable links, and the minimum required number of virtual channels. For instance, the *finter* [16] scheme using four VCs can support no more than two intermediate nodes.

Virtual channels can be very expensive to implement, requiring additional memory resources and associated allocation and arbitration logic. The TBOR algorithm has much more relaxed constraints with regard to the number of VCs. Even without a virtual channel, node pairs are able to communicate as long as there exists a connectable path in the AICDGs. Nevertheless, for better fault tolerance (Table 3) and network performance, the proposed algorithm uses

3. The combined turn models are consistent with the results shown in Fig. 6.

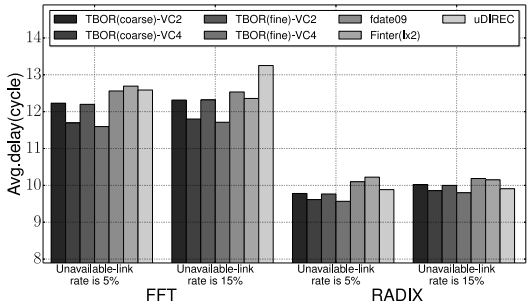


Fig. 8. Latency graphs for the FFT and RADIX SPLASH-2 benchmarks.

multiple VCs. In the fine-grained scheme, if a link is not physically broken or all its related crossbar switches are not damaged, then the link is deemed functional as long as one of its VCs is still operational. Therefore, using more VCs also decreases the possibility of out-of-service links caused by unavailable downstream VCs.

We evaluated the relationship between the unavailable-link rate and the reliability of the network by repeating the experiment $100,000$ times for each rate to explore as many different combinations as possible. The statistical results are shown in Table 3.The TBOR-fine method using two VCs shows very good reliability even with 30 percent unavailable links (34 out-of-service links of 112 total links). Only 8.9 percent of the simulations fail to complete under the UNIFORM-RANDOM traffic. For lower rates, a single turn model is sufficient to satisfy communication requirements for most cases. For instance, with 5 percent unavailable-link rate under the UNIFORM-RANDOM traffic, 92.18 and 93.51 percent of the simulations are handled without a virtual channel. Furthermore, applying the fine-grained scheme and using more VCs exhibit better results than applying the coarse-grained scheme and less virtual channels.

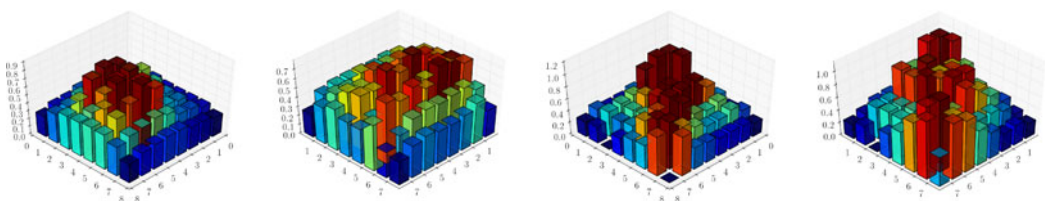
5.6 Power and Area Analysis

Link utilization can be used for high level power analysis for OCNs [59]. The amount of traffic or flits traversing the crossbar, being written into and being read out of virtual channels is fully captured in the link activity. Fig. 9 shows the spatial power distribution of the traffic load on the network. Cuboids of height zero indicate the position of unavailable nodes (TBOR using the coarse-grained scheme), while taller cuboids indicating higher link activities. Traffic loads are more evenly distributed across the network using the TBOR approach than the *finter(Ix2)* scheme. This is because the TBOR routing algorithm will choose non-minimal paths to avoid traffic congestion, making load balancing another key routing goal. Although approximations of the power utilization in networks are generally done based on traffic loads, routing algorithms cannot be overlooked. Fig. 10 using energy-delay product per flit (EDPPF) as the power-efficiency metric, shows that in most cases, the TBOR algorithm is a little better than the *finter(Ix2)* scheme, even though non-minimal routing is applied. Better traffic load-balancing leads to a less congested network which can result in overall lower latency per packet. The TBOR(fine) approach is better than the TBOR (coarse) method for the same reason.

Detecting faulty states in the on-chip network fabric is primarily the function of fault diagnosis schemes. Compared with the coarse-grained approach, the fine-grained diagnosis needs to specify the correct functionality of individual components and links. We applied the same mechanism as proposed by Kohler [17] for collecting detailed diagnostic information. Error detection units are needed at each input and output port. Crossbar faults can be managed by fault matrices, which are implemented as $in \times out$ register arrays. Link faults can be detected by comparing the two counters, one located at the input – recording the number of received packets and the other one located at the output – keeping

TABLE 3
Reliability with Increasing Number of Unavailable Links in an 8×8 2D Mesh Using TBOR

Unavailable link's rate	BIT-COMPLEMENT				TRANSPOSE				UNIFORM-RANDOM			
	Without VC		2 VCs		Without VC		2 VCs		Without VC		2 VCs	
	Coarse	Fine	Coarse	Fine	Coarse	Fine	Coarse	Fine	Coarse	Fine	Coarse	Fine
5%	96.82%	97.74%	99.65%	100%	99.79%	99.88%	100%	100%	92.18%	93.51%	99.72%	100%
10%	74.34%	88.63%	98.42%	99.67%	97.90%	99.23%	99.37%	99.36%	68.71%	81.81%	98.55%	99.67%
15%	43.62%	57.92%	94.39%	98.13%	94.02%	95.26%	96.67%	99.12%	37.78%	52.87%	93.78%	98.42%
20%	22.13%	45.46%	88.86%	96.73%	82.12%	93.32%	92.65%	98.84%	15.79%	38.83%	87.82%	97.48%
30%	4.80%	11.18%	66.01%	91.45%	44.83%	69.91%	74.31%	92.40%	1.82%	8.67%	67.71%	91.10%
40%	1.45%	3.90%	56.24%	88.10%	27.14%	48.10%	64.98%	92.00%	0.54%	2.64%	46.83%	85.29%



(a) Unavailable-link rate 5% uDRIEC (b) Unavailable-link rate 5% TBOR (c) Unavailable_link rate 15% uDIREC (d) Unavailable-link rate 15% TBOR

Fig. 9. Power distribution derived from HORNET [57] and DSENT [55] of an 8×8 2D network with four-VC per port. The buffer depth is 8 flits, the manufacture process technology is 45nm, and the frequency is 1 GHz. The benchmark in (a) and (b) is BIT-COMPLEMENT, the unavailable node and links are 35 and (13, 21), (48, 49); in (c) and (d) the TRANSPOSE traffic pattern is used, unavailable nodes and links are 23, 35, 52, and (3, 4), (5, 6), (17, 18), (25, 26), and (48, 49). Here we are showing the coarse-grained TBOR algorithm; other configurations exhibited the same pattern.

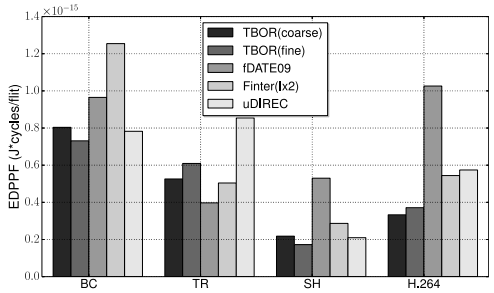


Fig. 10. Energy-delay product per flit derived from HORNET and DSENT, faulty or unavailable links rate is 10 percent. 10,000,000 flits were sent and delivered in each case.

track of the number of sent probes. As described in [17], the area overhead is less than 10 percent for the fine-grained scheme compared with the coarse-grained approach.

6 Conclusions

In this work, we develop an efficient approach for routing on-chip traffic around unavailable network resources, including topological irregularities caused component failures, oversized IP blocks and fine-grained ON/OFF operation of VFIs that significantly improves on-chip network routing performance. The key contributions of this work are:

1) Using a weighted acyclic channel dependency graph derived from the irregular network with one or more turn models and VC partitioning to overcome deadlock, load balancing, and resource under-utilization issues.
2) Proposing an algorithm for determining the minimum number of virtual channels needed to meet specific or universal application communication requirements.
3) Proposing both fine- and coarse-grained analysis schemes to achieve full utilization of network resources and to avoid the negative effect on network performance when a partially operational router is disabled completely.
4) Describing a novel traffic balancing oblivious routing algorithm named TBOR as a concrete implementation of the aforementioned techniques.

Experiments show that a fine-grained scheme, which treats a defective router as semi-faulty, significantly improves network performance and minimizes fault-induced loss of functionality- or processing- capability. The network fault diagnosis mechanism gives information on the network fault status. The fine-grained diagnosis method needs to investigate the fault status of each individual components of the router, which is much more expensive to implement than simply determining whether the router is defective. In our coarse-grained fault diagnosis approach, a defective router is classified as fully out-of-service to simplify design complexity. The version of the TBOR algorithm still improves throughput as well as the average flit latency over existing solutions. The TBOR algorithm uses fewer virtual channels while maintaining higher throughput than other algorithms. Additionally, in-order delivery is guaranteed by unsplit flow analysis and static virtual channel allocation.

Acknowledgments

The authors want to thank Mieszko Lis at the University of British Columbia and Srinivas Devadas at MIT for their comments and feedback on this work. This research is partially funded by key project of NSFC No. 61231018, NSFC grant No. 610303036, China Postdoctoral Science Foundation No. 2012M521777, National High Technology Research and Development Program of China No. 2014AA01A301, and the University of Oregon Division of Equity and Inclusion.

References

[1] S. Borkar, "Thousand core chips: A technology perspective," in *Proc. Des. Autom. Conf.*, 2007, pp. 746–749.
[2] R. Marculescu, U. Y.Ogras, L. shiuan Peh, N. E. Jerger, and Y. Hoskote, "Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 28, no. 1, pp. 3–21, Jan. 2009.
[3] S. Rusu, H. Muljono, D. Ayers, S. Tam, W. Chen, A. Martin, S. Li, S. Vora, R. Varada, and E. Wang, "5.4 Ivytown: A 22nm 15-core enterprise xeon processor family," in *Proc. IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2014, pp. 102–103.
[4] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. MacKay, M. Reif, L. Bao, J. Brown, M. Mattina, C. Miao, C. Ramey, D. Wentzlaff, W. Anderson, E. Berger, N. Fairbanks, D. Khan, F. Montenegro, J. Stickney, and J. Zook, "TILE64 - processor: A 64-Core SoC with mesh interconnect," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2008, pp. 88–598.
[5] A. Ivanov and G. D. Micheli, "The network-on-chip paradigm in practice and research," *IEEE Des. Test Comput.*, vol. 22, no. 5, pp. 399–403, 2005.
[6] S.-Y. Lin, C.-H. Huang, C.-H. Chao, K.-H. Huang, and A.-Y. Wu, "Traffic-balanced routing algorithm for irregular mesh-based On-chip networks," *IEEE Trans. Comput.*, vol. 57, no. 9, pp. 1156–1168, Sep. 2008.
[7] M. K. Schafer, T. Hollstein, H. Zimmer, and M. Glesner, "Deadlock-free routing and component placement for irregular mesh-based networks-on-chip," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2005, pp. 238–245.
[8] P. Teehan, M. Greenstreet, and G. Lemieux, "A survey and taxonomy of gals design styles," *IEEE Des. Test Comput.*, vol. 24, no. 5, pp. 418–428, Sep./Oct. 2007.
[9] M. H. Foroozannejad, M. Hashemi, A. Mahini, B. M. Baas, and S. Ghiasi, "Time-scalable mapping for circuit-switched GALS chip multiprocessor platforms," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 33, no. 5, pp. 752–762, May 2014.
[10] U. Y. Ogras, R. Marculescu, D. Marculescu, and E. G. Jung, "Design and management of voltage-frequency island partitioned networks-on-chip," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 17, no. 3, pp. 330–341, Mar. 2009.
[11] O. Ozturk, M. Kandemir, and G. Chen, "Compiler-directed energy reduction using dynamic voltage scaling and voltage islands for embedded systems," *IEEE Trans. Comput.*, vol. 62, no. 2, pp. 268–278, Feb. 2013.
[12] S. Borkar, "Designing reliable systems from unreliable components: The challenges of transistor variability and degradation," in *IEEE Micro*, vol. 25, no. 6, pp. 10–16, Nov./Dec. 2005.
[13] C. Nicopoulos, S. Srinivasan, A. Yanamandra, D. Park, V. Narayanan, C. Das, and M. Irwin, "On the effects of process variation in network-on-chip architectures," *IEEE Trans. Dependable Secure Comput.*, vol. 7, no. 3, pp. 240–254, Jul.–Sep. 2010.
[14] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. San Mateo, CA, USA: Morgan Kaufmann, 2004.
[15] J. Wu, "A fault-tolerant and deadlock-free routing protocol in 2d meshes based on odd-even turn model," *IEEE Trans. Comput.*, vol. 52, no. 9, pp. 1154–1169, Sep. 2003.
[16] M. Gomez, N. Nordbotten, J. Flich, P. Lopez, A. Robles, J. Duato, T. Skeie, and O. Lysne, "A routing methodology for achieving fault tolerance in direct networks," *IEEE Trans. Comput.*, vol. 55, no. 4, pp. 400–415, Apr. 2006.
[17] A. Kohler, G. Schley, and M. Radetzki, "Fault tolerant network on chip switching with graceful performance degradation," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 29, no. 6, pp. 883–896, Jun. 2010.

[18] K. Aisopos, C.-H. O. Chen, and L.-S. Peh, "Enabling system-level modeling of variation-induced faults in networks-on-chips," in *Proc. 48th Des. Autom. Conf.*, 2011, pp. 930–935.
[19] A. Banerjee and S. Moore, "Flow-aware allocation for on-chip networks," in *Proc. Int. Symp. Networks-on-Chip*, 2009, pp. 183–192.
[20] X. Zhong and V. Lo, "Application-specific deadlock free wormhole routing on multicomputers," in *Proc. 4th Int. PARLE Conf. Parallel Archit. Languages Europe*, 1992, pp. 193–208.
[21] C. J. Glass and L. M. Ni, "The turn model for adaptive routing," in *Proc. Int. Symp. Comput. Archit.*, 1992, pp. 278–287.
[22] G. Chiu, "The odd-even turn model for adaptive routing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 11, no. 7, pp. 729–738, Jul. 2000.
[23] A. Shafiee, M. Zolghadr, M. Arjomand, and H. Sarbazi-Azad, "Application-aware deadlock-free oblivious routing based on extended turn-model," in *Proc. Int. Conf. Comput.-Aided Des.*, 2011, pp. 213–218.
[24] B. Fu, Y. Han, J. Ma, H. Li, and X. Li, "An abacus turn model for time/space-efficient reconfigurable routing," in *Proc. 38th Annu. Int. Symp. Computer Archit.*, 2011, pp. 259–270.
[25] J. Stankovic, M. Spuri, M. Di Natale, and G. Buttazzo, "Implications of classical scheduling results for real-time systems," *IEEE Comput.*, vol. 28, no. 6, pp. 16–25, Jun. 1995.
[26] V. Lo, "Heuristic algorithms for task assignment in distributed systems," *IEEE Trans. Comput.*, vol. 37, no. 11, pp. 1384–1397, Nov. 1988.
[27] B. Towles and W. Dally, "Worst-case traffic for oblivious routing functions," in *Proc. 14th Annu. ACM Symp. Parallel Algorithms Archit.*, 2002, pp. 1–8.
[28] W. Dally and C. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," in *IEEE Trans. Comput.*, vol. C-36, no. 5, pp. 547–553, May 1987.
[29] M. A. Kinsy, M. H. Cho, T. Wen, E. Suh, M. van Dijk, and S. Devadas, "Application-aware deadlock-free oblivious routing," in *Proc. Int. Symp. Comput. Archit.*, 2009, pp. 208–219.
[30] M. H. Cho, C.-C. Cheng, M. Kinsy, G. E. Suh, and S. Devadas, "Diastolic arrays: Throughput-driven reconfigurable computing," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2008, pp. 457–464.
[31] T. Nesson and S. L. Johnsson, "ROMM routing: A class of efficient minimal routing algorithms," in *Proc. Int. Workshop Parallel Comput. Routing Commun.*, 1994, pp. 185–199.
[32] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2001.
[33] K. Walkowiak, "New algorithms for the unsplittable flow problem," in *Proc. Int. Conf. Comput. Sci. Its Appl.*, 2006, pp. 1101–1110.
[34] K. S. Shim, M. H. Cho, M. A. Kinsy, T. Wen, M. Lis, E. Suh, and S. Devadas, "Static virtual channel allocation in oblivious routing," in *Proc. Int. Symp. Networks-on-Chip*, 2009, pp. 38–43.
[35] U. Y. Ogras, R. Marculescu, P. Choudhary, and D. Marculescu, "Voltage-frequency island partitioning for GALS-based networks-on-chip," in *Proc. 44th ACM/IEEE Des. Autom. Conf.*, 2007, pp. 110–115.
[36] A. Chien and J. Kim, "Planar-adaptive routing: Low-cost adaptive networks for multiprocessors," *ACM SIGARCH Comput. Archit. News*, vol. 20, no. 2, pp. 268–277, 1992.
[37] C. Glass and L. Ni, "Fault-tolerant wormhole routing in meshes," in *Proc. 23rd Int. Symp. Fault-Tolerant Comput. Dig. Papers.*, 1993, pp. 240–249.
[38] R. Boppana and S. Chalasani, "Fault-tolerant wormhole routing algorithms for mesh networks," *IEEE Trans. Comput.*, vol. 44, no. 7, pp. 848–864, Jul. 1995.
[39] C. Chen and G. Chiu, "A fault-tolerant routing scheme for meshes with nonconvex faults," *IEEE Tran. Parallel Distrib. Syst.*, vol. 12, no. 5, pp. 467–475, May 2001.
[40] J. Duato, "A necessary and sufficient condition for deadlock-free adaptive routing in wormhole networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 6, no. 10, pp. 1055–1067, Oct. 1995.
[41] K. Gopalan, T.-c. Chiueh, and Y.-J. Lin, "Load balancing routing of fault tolerant QoS-guaranteed VPNs," in *Proc. 15th IEEE Int. Workshop Quality of Service*, 2007, pp. 100–108.
[42] D. Fick, A. DeOrio, G. Chen, V. Bertacco, D. Sylvester, and D. Blaauw, "A highly resilient routing algorithm for fault-tolerant NoCs," in *Proc. Conf. Des., Autom. Test Eur. Conf. Exhib.*, 2009, pp. 21–26.
[43] M. Ebrahimi, M. Daneshtalab, J. Plosila, and F. Mehdipour, "MD: Minimal path-based fault-tolerant routing in on-chip networks," in *Proc. Des. Autom. Conf. 18th Asia South Pacific*, 2013, pp. 35–40.
[44] M. Ebrahimi, M. Daneshtalab, J. Plosila, and H. Tenhunen, "Minimal-path fault-tolerant approach using connection-retaining structure in networks-on-chip," in *Proc. 7th IEEE/ACM Int. Symp. Netw. Chip*, 2013, pp. 1–8.
[45] M. Ebrahimi, M. Daneshtalab, F. Farahnakian, J. Plosila, P. Liljeberg, M. Palesi, and H. Tenhunen, "HARAQ: Congestion-aware learning model for highly adaptive routing algorithm in on-chip networks," in *Proc. 6th IEEE/ACM Int. Symp. Netw. Chip.*, 2012, pp. 19–26.
[46] C. Feng, Z. Lu, A. Jantsch, J. Li, and M. Zhang, "A reconfigurable fault-tolerant deflection routing algorithm based on reinforcement learning for network-on-chip," in *Proc. 3rd Int. Workshop Netw. Chip Archit.*, 2010, pp. 11–16.
[47] P. Ren, Q. Meng, X. Ren, and N. Zheng, "Fault-tolerant routing for on-chip network without using virtual channels," in *Proc. 51st Annu. Des. Autom. Conf. Des. Autom. Conf.*, 2014, pp. 1–6.
[48] D. Xiang, "Deadlock-free adaptive routing in meshes with fault-tolerance ability based on channel overlapping," *IEEE Trans. Dependable Secure Comput.*, vol. 8, no. 1, pp. 74–88, Jan./Feb. 2011.
[49] P. Lotfi-Kamran, M. Daneshtalab, C. Lucas, and Z. Navabi, "Barp-a dynamic routing protocol for balanced distribution of traffic in nocs," in *Proc. Conf. Des., Autom. Test Eur.*, 2008, pp. 1408–1413.
[50] M. Palesi and M. Daneshtalab, *Routing Algorithms in Networks-on-Chip*. New York, NY, USA: Springer, 2014.
[51] T. Xanthopoulos, "Fault tolerant adaptive routing in multicomputer networks," Ph.D. dissertation, California Inst. Technol., Pasadena, CA, USA, 1995.
[52] D. Linder and J. Harden, "An adaptive and fault tolerant wormhole routing strategy for k-ary n-cubes," in *IEEE Trans. Comput.*, vol. 40, no. 1, pp. 2–12, Jan. 1991.
[53] R. Parikh and V. Bertacco, "uDIREC: Unified diagnosis and reconfiguration for frugal bypass of NoC faults," in *Proc. 46th Annu. IEEE/ACM Int. Symp. Microarchit.*, 2013, pp. 148–159.
[54] P. Ren, M. Lis, M. H. Cho, K. S. Shim, C. W. Fletcher, O. Khan, N. Zheng, and S. Devadas, "HORNET: A cycle-level multicore simulator," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 31, no. 6, pp. 890–903, Jun. 2012.
[55] C. Sun, C.-H. O. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L.-S. Peh, and V. Stojanovic, "DSENT—A tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling," in *Proc. 6th ACM/IEEE Int. Symp. Netw.-on-Chip*, 2012, pp. 201–210.
[56] K. Skandron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," in *Proc. Int. Symp. Comput. Archit.*, 2003, pp. 2–13.
[57] M. Lis, P. Ren, M. Cho, K. Shim, C. Fletcher, O. Khan, and S. Devadas, "Scalable, accurate multicore simulation in the 1000-core era," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw.*, 2011, pp. 175–185.
[58] W. Dally and H. Aoki, "Deadlock-free adaptive routing in multicomputer networks using virtual channels," *IEEE Trans. Parallel Distrib. Syst.*, vol. 4, no. 4, pp. 466–475, Apr. 1993.
[59] N. Eisley and L. Peh, "High-level power analysis for on-chip networks," in *Proc. Int. Conf. Compilers, Archit. Synthesis Embedded Syst.*, 2004, pp. 104–115.

Pengju Ren received the PhD degree in electrical engineering from Xi'an Jiaotong University in 2012. He is an assistant professor in the School of Electronic and information Engineering, Xi'an Jiaotong University. During 2009 to 2011, he was a visiting PhD student in computer science and Artifical Intelligence Laboratory in Massachusetts Institute of Technology (MIT). His current research interests include On-chip network, scalable many-core designs and VLSI architecture for digital video processing. He is a member of the IEEE.

Michel A. Kinsy received the BSE degree in computer systems engineering and the BS in computer science, both from Arizona State University, the MS degree in electrical engineering and computer science from MIT, and the PhD degree in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT) in 2013. He is an assistant professor in the Department of Computer and Information, University of Oregon and the director in the Computer Architecture and Embedded Systems (CAES) Laboratory. He is an MIT presidential fellow. His research interests include cognitive high-performance many-core architectures, self-aware adaptive multicore systems, intelligent network-on-chip designs, hardware and embedded systems security, and cyberphysical systems. He is a member of the IEEE.

Nanning Zheng graduated from the Department of Electrical Engineering, Xi'an Jiaotong University, Xi'an, China, in 1975, and received the PhD degree in electrical engineering from Keio University, Yokohama, Japan, in 1985. He is currently a professor and the director in the Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University. His research interests include computer vision, pattern recognition, machine vision and image processing, neural networks, and hardware implementation of intelligent systems. He became a member of the Chinese Academy of Engineering in 1999, and he is the Chinese Representative on the Governing Board of the International Association for Pattern Recognition. He also serves as an executive deputy editor in the Chinese Science Bulletin. He is a fellow of the IEEE.