Dodec: Random-Link, Low-Radix On-Chip Networks

Haofan Yang, Jyoti Tripathi, Natalie Enright Jerger
Edward S. Rogers Department of Electrical and Computer Engineering
University of Toronto, Toronto, Canada
{*haofan.yang,jyoti.tripathi*}*@mail.utoronto.ca, enright@ece.utoronto.ca*

Dan Gibson
Google, Inc.
Madison, WI USA
deg@google.com

***Abstract*—Network topology plays a vital role in chip design; it largely determines network cost (power and area) and significantly impacts communication performance in many-core architectures. Conventional topologies such as a 2D mesh have drawbacks including high diameter as the network scales and poor load balancing for the center nodes. We propose a methodology to design random topologies for on-chip networks. Random topologies provide better scalability in terms of network diameter and provide inherent load balancing. As a proof-of-concept for random on-chip topologies, we explore a novel set of networks – dodecs – and illustrate how they reduce network diameter with randomized low-radix router connections. While a 4×4 mesh has a diameter of 6, our dodec has a diameter of 4 with lower cost. By introducing randomness, dodec networks exhibit more uniform message latency. By using low-radix routers, dodec networks simplify the router microarchitecture and attain 20% area and 22% power reduction compared to mesh routers while delivering the same overall application performance for PARSEC.**

I. Introduction

Networks and communication play an increasingly important role in overall performance of many-core chip multiprocessors (CMPs). Topology bounds critical network metrics, such as latency, throughput and energy consumption. By affecting chip area and wiring complexity, the topology has a significant impact on the network implementation cost. On-chip networks (OCNs) should be designed with *short diameter* and *low radix*. Diameter[1] directly impacts *performance*, e.g., every OCN hop is potentially on the critical path of a load miss. Lower-diameter OCNs reduce the hop count required to service a request, which improves latency (and performance). Low-radix[2] routers are desirable due to their lower *cost* (power, area) arising from simpler router implementations.

A significant fraction of OCN research focuses on network optimizations using low-radix topologies such as meshes or tori [2], [3], [4]. These topologies are popular for their simplicity and ease of VLSI layout. Despite the range of optimizations proposed, the limitations of these regular, rigid topologies, such as a long network diameter, low path diversity and poor load balancing, impede further OCN performance improvements. Alternatively, high-radix routers [5] can deliver high network performance. However, router complexity increases quadratically impacting area and cost; the increase in radix also negatively impacts the router cycle time.

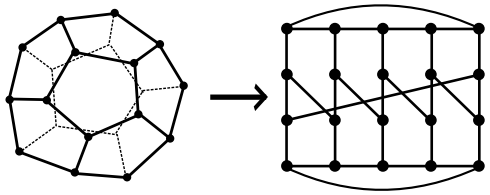


Figure 1: Dodecahedron and a sample planar topology

Our goal is to marry low-diameter networks and low-radix design through randomized link connections. Randomization has well-known, fundamental and desirable properties for networks including low diameter and balancing of worst-case communication patterns as seen with Valiant's randomized routing [6]. Simplicity makes low-radix routers attractive; however, meshes fail to effectively exploit their resources to reduce diameter. The regularity of a mesh limits the per-node reach of the network. Our proposed dodec networks[3] exploit the simplicity of *low-radix* routers by considering a uniform router radix of 3 throughout the network (one more than a ring) while achieving a *short diameter* through non-uniform connections resulting in a low average hop count. Fig. 1 shows a dodecahedron with 20 vertices and a possible planar layout of this 4×5 array. As described in Sec. III, our dodec OCN is not limited to 20 nodes; we generalize a dodecahedron to create a class of topologies.

This paper makes the following contributions:

- Illustrates how existing low-radix topologies fail to use resources effectively to reduce diameter;
- Explores the benefits of randomized link connections to achieve both short diameter and router simplicity;
- Develops a methodology for constructing random topologies and their required routing functions;
- Demonstrates that our proof-of-concept dodecs are more efficient than a mesh overall: At cost-parity, dodecs increase throughput by up to 50% and reduce mean

[1]Diameter defines the maximum distance between two nodes [1].

[2]The radix or degree of the router defines the number of ports per router.

[3]Our topology's name is inspired by a dodecahedron which has 3 edges/vertex, 12 flat faces and 20 vertices.

latency by 10%. At performance-parity, dodecs save 20% router area and 15% network power.

II. Motivation

Across a range of workloads, OCNs are often lightly loaded [7]. OCNs typically operate well below saturation. As a result, average OCN latency is close to the zero-load latency. Zero-load latency consists of head latency and serialization latency. Head latency is the time the head flit[4] spends traversing the network and is a function of the hop count. Serialization latency is the time for the remaining flits to reach the destination and is a function of the channel bandwidth. Assuming equal per-router bandwidth, we see a trade-off between head and serialization latencies when choosing between high- and low-radix OCNs. In a high-radix OCN, head latency shrinks as hop count decreases, while serialization latency increases due to the decrease in per-channel bandwidth as more ports are added. However, as radix increases so does the complexity of the router microarchitecture. Low-radix OCNs enjoy the benefits of simple routers but suffer from long network diameters which results in high average hop counts. For each hop, a packet pays router pipeline latency and link traversal latency. As router latency is paid on a per-hop basis, it has a significant impact on overall latency and has been the focus of numerous pipeline optimizations. Extra hops also lead to higher energy consumption as router traversals contribute significantly to total network energy [8].

The network diameter depends not only on *radix*, but also on *topology*, i.e., how routers connect to each other. High-radix networks reduce network diameter by providing each node with abundant direct connections to other nodes, i.e., they give rich first-degree connections to every node. With rich first-degree connectivity, a given source node is able to reach a large number of nodes within two hops, i.e., second-degree connections. As a result, in a high-radix network, node coverage from the first few degrees of connections is usually large enough to cover the entire network. Dramatic improvements can be made in low-radix networks by focusing on reducing the average hop count by increasing node reach. Most traditional low-radix networks fail to achieve this due to restrictive topological constraints. To maintain regularity in the topology, many available connections are spent on interconnecting nodes within the same degree connection coverage. Take a mesh as an example (Fig. 2): for a given node (e.g. "S"), the first-degree connections cover 4 adjacent nodes. Given the constraint of a radix of 4,[5] the maximal second-degree connections could reach up to 12 nodes, but for a mesh, there are only 8 nodes within the second-degree connections. By simply altering one connection of that node, we increase the second-degree connection coverage to 9 nodes.

[4]A packet is subdivided into flits. In OCNs, flit size usually equals the channel width.

[5]Router radix does not count the injection/ejection port since it does not contribute to forming the topology.

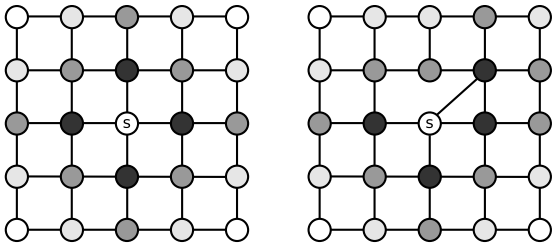


Figure 2: Impact of an irregular link on hop count

This example intuitively illustrates trading off network regularity for lower diameter. In this way, a low-radix network can reduce the average hop count, thus reducing the head latency, while maintaining high channel bandwidth to avoid high serialization latency. However, trade-offs come with these improvements. Irregular connections may be longer as they traverse multiple tiles; we will demonstrate that longer link latency with fewer hop traversals is a net win for performance. Irregular topologies also present challenges for routing and deadlock avoidance. We discuss these issues in later sections.

III. Polyhedral On-Chip Network Topologies

In this section, we present our methodology for designing an OCN with randomized connections. We focus on dodec topologies built out of radix-3 routers; however, the polyhedron methodology can be applied to routers of any degree. Our methodology requires 3 steps: connectivity graph generation, placement optimization and routing computation. Our focus is the design and methodology; optimizations to improve algorithmic scalability are straightforward.

Connectivity Graph Generation. First, we create a topological connectivity graph that specifies each node's connectivity. For an N-node polyhedron, the total number of channels (N_c) is $\frac{r \times N}{2}$, where r is the router radix. A topological connectivity graph is created by establishing connections from each node to r other random nodes. The process is successful only when all nodes form a single connected graph, otherwise, the algorithm aborts and restarts until it finds a legal connectivity graph. Because many legal dodec configurations provide good average distance for all nodes, an architect has freedom in the design space to apply customized changes to fit requirements. For example, in SoC design we would want IP blocks that have frequent communication to be close to each other. Distance weighting can be applied to achieve this type of optimization.

Placement Optimization. Mapping an arbitrary polyhedron topology to a planar VLSI layout presents several challenges. Poor node placement can lead to link asymmetry, adversarial thermal hotspots, and verification complexity. A key strength of polyhedron networks is that they afford flexibility for the

circuit implementation. E.g., in Sec. VI-B, we demonstrate how the dodec topology can take advantage of available on-chip wiring to mitigate circuit implementation concerns.

Node placement lends itself well to existing placement heuristics. We use a genetic algorithm [9] to optimize placement, with the target of minimizing average link length under uniform link length weighting. We use a fixed-effort configuration of 1000 parents for 100 iterations with a new solution rate of 500 crossovers per generation. Each potential placement is evaluated according to a combined fitness score considering link length as a metric. For simplicity, we considered placements of tiles of fixed size and geometry; this assumption is not fundamental.

Routing Computation. Routing in a polyhedron network is challenging due to the irregular layout. For this reason, we use table-based routing. The routing paths are computed offline after the 2-D planar placement optimization and stored in per-node routing tables. The table-based routing mechanism also provides significant room for customized design. Routing is discussed in detail in Sec. IV.

A. Proof-of-Concept: Dodec

Considering the trade-offs discussed in Sec. II, we explore building a low-radix topology with a low network diameter through the use of randomized connections. Although our methodology is general, we focus on a specific radix to demonstrate the performance and opportunity for randomized polyhedron OCNs. We select *radix of 3* as it is the smallest radix that can form different topologies. A 3-radix router is desireable because it represents a simplified design with respect to more canonical router microarchitectures (i.e., those of radix 4). Such an architecture may not be feasible for high-radix routers [10], whereas a small radix leads to better performance and lower area.

Our dodec topology is inspired by a class of polyhedra known as dodecahedra, which have 12 flat faces and 20 vertices (Fig. 1 left). However, the broader class of dodecs can generalize to any network with an even number of nodes. All vertices in a dodecahedron uniformly have a radix of 3, and flattening the dodecahedron to a planar space creates a new 3-radix topology. We find average hop count of this newly created topology is commonly low, demonstrating both lower potential cost and promising good OCN performance.

Dodec Topology Description. Dodec exploits trade-offs between regularity and diameter while enjoying the benefits of ultra low-radix (3) routers. To avoid increasing the network diameter, the dodec network relaxes constraints of regularity to achieve higher node connectivity. The dodec networks define a set of networks that conform to the same constraints, and thus have some common characteristics. We define the dodec topology formally:

Definition. *A dodec network topology is a connected topology comprised of vertices with precisely three edges, each connected to different neighbors.*

Network Diameter	3	4	5	6	7	8
Frequency	1	577	377	37	7	1

Table I: Diameter frequency of 1000 random 16-node dodecs

Link Length	1	2	3
Percentage	36%	41%	23%

Table II: Link lengths of 1000 random 16-node dodecs

By placing no constraints on how nodes connect to each other, the dodec network embraces all possible connection patterns to reduce network diameter, thus enabling broad opportunities for customizing system design. To demonstrate the similarity of different dodecs, we generate 1000 instances of dodec networks. Table I shows the frequency that different network diameters are realized. The majority of random dodecs achieve a network diameter of 4 with one diameter-3 dodec network. A 16-node mesh and ring have diameters of 6 and 8. Fig. 3 shows a distribution of the average hop count for these 1000 dodec instances. The overwhelming majority of dodecs have a lower average hop count than a 16-node mesh which has an average hop count of 2.67.

When mapping a dodec network onto a typical VLSI planar layout, some links are longer in order to traverse multiple nodes. Since on-chip wire latency is proportional to its Manhattan distance, the longer dodec links consume multiple link traversal cycles. We apply our placement optimization to reduce the link length. Table II shows that after the placement optimization the longest link length in a 16-node dodec is only 3 (the longest link prior to optimization is 6). Fig. 4 shows a typical dodec network and its optimized planar placement.

Extra link latency contributes an acceptable overhead to the network and is compensated for by the lower average hop count. According to Table II, there only a few length-3 links. Extra link latency only impacts packets traveling across the long link (by 1-2 cycles in a 16-node dodec); it will not impact the router pipeline or frequency.[6] Link length is sacrificed to reduce hop count. A smaller hop count reduces head latency by the unit of per router latency. Given that router pipeline latency dominates in canonical implementations,[7] the reduction in latency is significant, as it is worthwhile to trade-off link complexity for a simpler router microarchitecture. We explore layout constraints in Sec. VI-B.

IV. Routing Algorithms

Calculating routes in a polyhedron could be as simple as conducting a shortest path search for each source-destination pair. Given the irregularity of polyhedron OCNs, this approach is insufficient to prevent potential deadlock. Both node placement and routing optimization can vary significantly for

[6]We discuss the impact of extra link latency and power in Section V.
[7]Router pipelines typically have 3-5 stages.

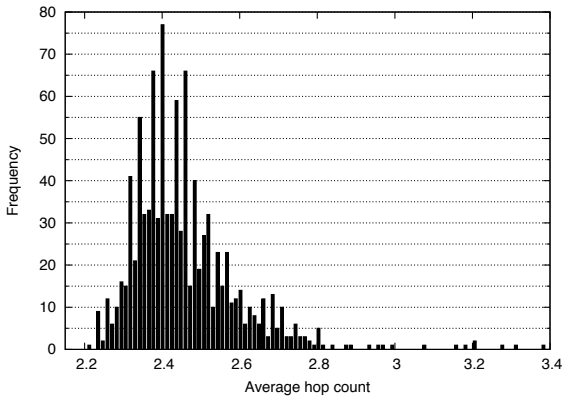


Figure 3: Average hop count of 1000 random 16-node dodecs

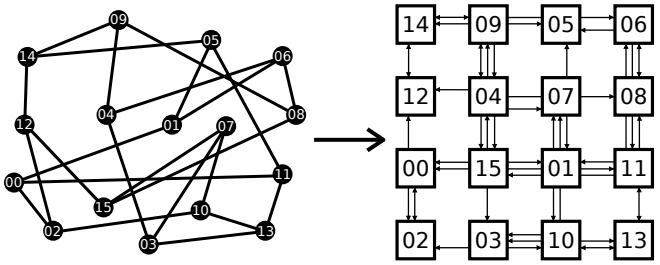


Figure 4: An exemplary dodec and its 2D planar placement

different design targets. Just as the placement fitness function can be tailored to a designer's performance requirements, so can the generation of the routing algorithm. As a proof-of-concept for polyhedral networks, we focus on obtaining a routing algorithm that achieves good overall performance.

A. Channel Dependency and Deadlock Avoidance

Deadlock occurs when a cyclic relationship forms between packets occupying resources while requesting other resources in order to move forward; due to this cycle, no packet can make progress. To design a deadlock-free routing algorithm, a channel dependency graph (CDG) is constructed by analyzing the channel resource dependences [11]. In this directed graph, a vertex represents each resource and the directed edge between two vertices denotes a possible dependency between those resources. To avoid deadlock, all cycles in the dependence graph must be eliminated through flow control or the routing algorithm. In a topology such as a mesh, channel dependences can be eliminated through routing turn restrictions between dimensions. Turn elimination yields an acyclic CDG graph. *Dimension Order Routing* is one example of a deadlock-free routing function.

Deadlock-free Routing on Polyhedral OCNs. We apply a turn model strategy to polyhedron. We refer to a *turn* as an input-output channel pair that a packet might traverse within a router. Each turn represents a dependency edge in the CDG. Consider the simple, 4-node dodec in Fig. 5 (left), assuming all ports are available to all inputs, we derive its CDG as

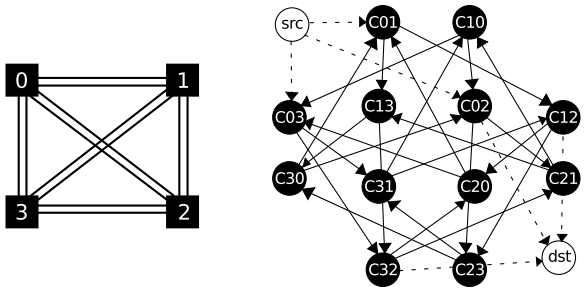


Figure 5: Example dodec (left) and its channel dependency graph (right). Channels are named $C + src\# + dst\#$; e.g., channel connecting node 0 to node 1 is C01.

shown in Fig. 5 (right). The directed edge connecting vertex `C01` to `C12` represents a dependency between the west input and the south output of node 1. By regulating turns in each router, we form an acyclic CDG. We use depth-first search (DFS) and randomization to eliminate dependency cycles in the CDG. Branches that result in a disconnected network are discarded. Randomized search and elimination is sufficient for 16-64 nodes scale we focus on. There are only a few ($<$10) cycles in a 16-node dodec. Search complexity is an issue for larger numbers of nodes but optimizations to quickly generate a large acyclic CDG are outside of the scope of this paper. Based on the acyclic CDG, we derive routes that connect source and destination nodes [12]: We add additional pseudo source and destination vertices to find the shortest path from source to destination. Fig. 5 shows how pseudo vertices are added to compute routes from node 0 to node 2 based on the permitted turns in the CDG.

Since the network connections are randomized, it is sufficient to randomize the selection of forbidden turns to attain a load-balanced routing function. Different acyclic CDGs may result in deadlock-free routes of different qualities; however, we found that the performance difference is minor. Designers can impose customized constraints to build their OCN. As some combinations of forbidden turns can result in node disconnection in the network, one may need several iterations to maintain a connected routing function.

To implement turn restrictions via table-based routing, the routing function (R) needs to define the relationship between incoming and outgoing channels. The packet's output port is calculated at each node based on the destination (N_{dest}) and the packet's input port (C_{in}). N_{dest} and C_{in} are indices into per-node routing tables. In a 16-node dodec, 64 2-bit entries in each router store our deterministic routing function. 2 bits encode 4 outgoing channels. A 128-bit routing table is small relative to the input buffering in the routers; its access latency does not impact the critical path through the router.

Adaptive Routing. Deadlock-free routing is crucial because it guarantees correct functionality under worst-case scenarios. However, some routes calculated from the CDG are non-minimal which increases the average hop count and hurts

Name	Routing
dodec	adaptive deadlock-free routing
mesh-dim	dimensional order routing
mesh-romm	ROMM [16]
torus/ring-dim	dimensional order routing
torus/ring-adp	minimal adaptive routing [17]

Table III: Topologies & routing algorithms for comparison

network performance. The average hop count for a typical 16-node dodec increases from 2.32 to 2.75 when using deadlock-free routes. Moreover, the deadlock-free routing function cannot exploit path diversity for load balancing. To improve performance, we combine a minimal routing function and a deadlock-free routing function to create an adaptive routing function. We use *escape virtual channels* (EVCs) [13], [14] for deadlock avoidance. We assign the previously discussed deadlock-free routing function to the EVCs. If deadlock occurs on the adaptively-routed VCs (normal VCs), a packet can escape to an EVC and follow a deterministic, deadlock-free path. We implement a fully-adaptive minimal routing algorithm by returning multiple legal output channels when multiple minimal paths exist and assign them to normal VCs. By leveraging EVCs, we improve overall network performance.

Additional considerations are needed when combining deadlock-free and deadlock-prone routing functions. Using wormhole flow control, packets traversing between a normal VC and an EVC create an *indirect channel dependency* [13]. We resolve this problem by breaking these indirect channel dependency cycles. Packets in normal VCs can select between both adaptive and deadlock-free routes. However, once a packet enters an EVC, it must remain on the deadlock-free route using EVCs until reaching its destination. While this approach is intuitive, it can lead to performance bottlenecks; as packets can never "escape" from EVCs, EVCs are forced to absorb most traffic. To avoid a disproportionate load on the EVCs, additional rules are applied [15]: (1) injected traffic is always assigned to a normal VC; (2) normal VCs are selected as the output with higher priority. These two rules ensure that packets only use EVCs when the network is congested.

The adaptive deadlock-free algorithm only needs minor changes to the routing tables. Instead of returning a single output channel for the deterministic deadlock-free algorithm, the routing table for the adaptive algorithm will return multiple results from both sub-routing algorithms. The routing table requires extra entries; with one alternative route for every input port-destination pair in a 16-node dodec, we only need an extra 64 2-bit entries in each router to store the additional information. Based on the output channel results, simple routing computation logic will determine which output channel(s) to apply for in VC allocation.

V. Evaluation

The design space of polyhedral networks generated through random connections is large. We focus our evaluation on

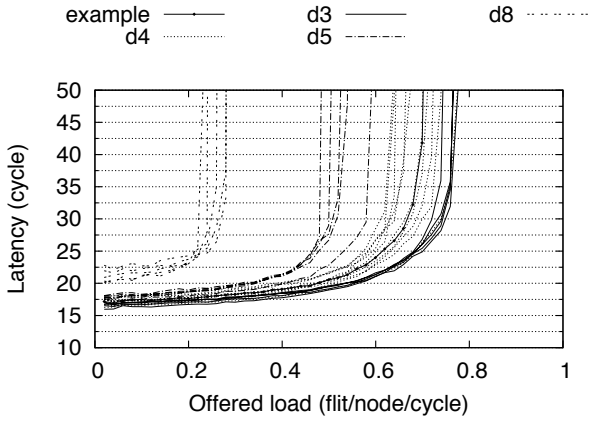


Figure 6: Performance of different dodec instances

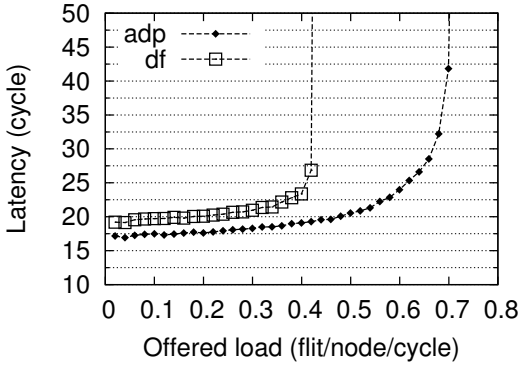
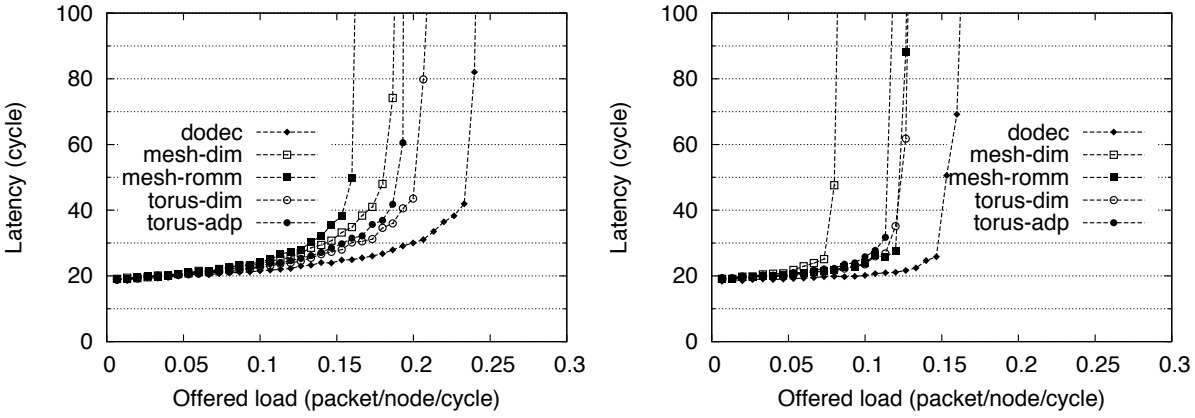


Figure 7: Routing comparison on example dodec network

our proof-of-concept dodec to demonstrate that randomized connections can mitigate the limitations commonly associated with low-radix networks. To present the general characteristics of dodecs, we evaluate different dodec instances, and choose a common-case 16-node dodec with diameter of 4 as a representative example. Based on this example network, we evaluate different routing functions. We modify Booksim, a cycle-accurate simulator [18] to model our dodec network. We use a canonical 3-stage pipelined VC router with speculation. We use 4 VCs with 8 flit buffers per VC. The pipeline and flow control are common across all networks. Multiple VCs are used to break both message- and protocol-level deadlock. Additional delay for long links is accurately modeled; a link that spans 1 tile consumes 1 cycle. When a link spans 2 or 3 tiles, it requires 2 or 3 cycles. Data is measured after the network is warmed up [1]. We compare against popular low-radix OCN topologies: ring, mesh and folded torus using synthetic traffic patterns. We run full-system simulations for PARSEC [19] using Booksim with FeS2 [20]. We run 16 cores with 2 levels of cache and a MOESI coherence protocol. We also consider 64-node networks and compare our proposal against flattened butterfly and concentrated mesh in Sec. VI-A.



(a) benign (b) adversarial

Figure 8: (a) benign and (b) adversarial traffic performance.

A. Performance of Different Dodec Networks

Fig. 6 explores the impact of diameter of network performance. We generate dodec instances with various network diameters and compare their throughput for single-flit uniform random traffic (dodecs with diameter ρ are denoted as `d`ρ). The performance curve of different network instances with the same diameter tend to be grouped together. As the diameter increases, the performance of each diameter group degrades; the groups with the longest and shortest diameters serve as performance lower and upper bounds. The most populous group has a diameter of 4, these designs perform similarly to the group with a rarer diameter of 3. This result is promising since it shows a large design space for network designers to customize different dodecs; customization may necessitate exploration of other evaluation metrics. To ensure a representative evaluation, we choose a dodec instance with medium performance (Fig. 4) from the most common group with a diameter of 4 (denoted as `example` in Fig. 6).

B. Routing Algorithm

Now that we have selected a representative dodec, we evaluate the non-minimal deadlock-free (`df`) and the adaptive (with 1 EVC) deadlock-free (`adp` algorithms discussed in Sec. IV. Fig 7 shows the performance of different routing algorithms under uniform random traffic. `Df` sacrifices throughput and zero-load latency for deadlock freedom. Non-minimal routes lead to higher zero-load latency. `Adp` performs better because normal VCs have higher priority so packets use the minimal routing function most of the time under light traffic loads. As a result, we use `adp` for further dodec evaluations.

In the deadlock-free routing generation process, different selections for eliminating channel dependency cycles may produce routing functions with different performance. We evaluate the impact of this variation by generating several different routing functions on the example dodec. In general, variations in routing algorithms have only a minor impact on network throughput (less than 7% difference) and no perceivable impact on zero-load latency. Topology generation rather than routing generation has a more significant impact.

C. Throughput Comparison

Fig. 8 compares the example 16-node dodec with regular topologies that have 1 more radix (mesh/torus) with deterministic and non-deterministic routing algorithms listed in Table III. Generally, there are more channels across the bisection in a dodec because it tries to reduce network diameter. If we assume a unit bandwidth for each channel in the networks presented, the bisection bandwidth for the mesh, example dodec and torus are 4, 6 and 8. Since different dodec instances may have different bisection bandwidth, holding bisection bandwidth constant is not the best point for comparison. Instead, to evaluate the trade-off between the number of router ports and channel bandwidth, we hold the total amount of outgoing channel bandwidth per-router constant, which is independent of placement. Results from holding bisection bandwidth constant across different networks show similar trends to the results in this section. We assume fixed equal-size packets; the packet consists of 3 and 4 flits in dodec and mesh/torus. We use both benign (uniform random) and adversarial (bit-reverse) traffic patterns. Adversarial patterns are dependent on topology; bit-reverse is adversarial for a mesh network.

With smaller diameter and higher per-channel bandwidth, the example dodec provides up to a 50% throughput increase compared to the mesh in Fig 8. It is important to point out that the performance of this dodec instance ranks in the middle of the 16-node dodec family, thus this evaluation understates the peak performance of a dodec topology. We measure latency in cycles; since the simple router in dodec gives us the opportunity for a higher clock frequency, the dodec also has an advantage in latency comparisons.

The worst adversarial traffic pattern may vary for different dodecs due to different bisection cuts. We design a custom

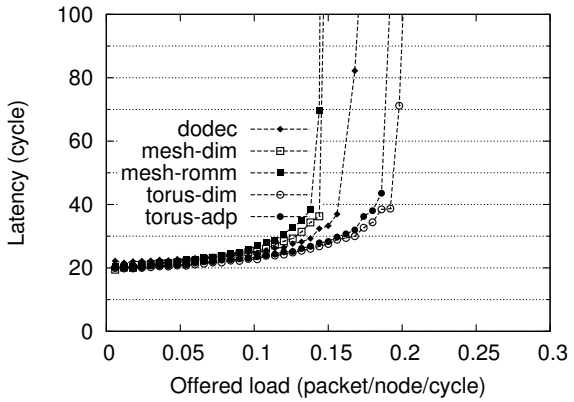


Figure 9: Dodec-adversarial traffic pattern

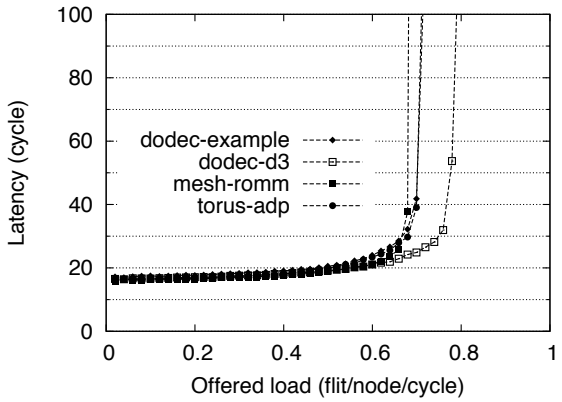


Figure 10: Performance under equal channel bandwidth

traffic pattern that represents a pathological case for the sample dodec. We determine the bisection cut for the sample dodec, and then let every source node send traffic only to random nodes in the other bisection partition. This pattern heavily loads the dodec's bisection channels. Fig. 9 compares the performance of all networks under this traffic pattern. Compared to Fig. 8 (bit-reverse), the dodec has nearly the same saturation throughput for adversarial traffic. While the torus performs better under this custom traffic pattern, the mesh suffers with this adversarial traffic pattern. This result reinforces the benefit of randomized connections in dodec which lead to less performance variation across different traffic patterns.

A ring suffers from high zero-load latency and relatively poor throughput due to long diameter and lack of path diversity (results omitted for space). Fig. 10 compares our dodec to a mesh and torus but with equal-per-channel bandwidth instead of the equal-per-router bandwidth (equal cost) used in the previous evaluation. Although zero-load latency varies with topology in Fig. 8 and 10, the difference is minor due to similar average hop counts. Single-flit packets are used because per-channel bandwidth is equal for all networks. Sec. VI-B shows the potential area savings of the dodec router in this scenario; reducing the number of ports while not increasing the per-channel bandwidth achieves area savings. Fig. 10 shows that the 3 topologies have nearly identical zero-load latencies and throughputs; at lower cost, a dodec can deliver performance equal to a more expensive mesh/torus. Choosing a dodec with diameter 3 (`dodec-d3`) increases saturation throughput; the cost of the diameter 3 dodec is equal to `dodec-example` so this improvement comes for free.

D. Synthetic Traffic Patterns

In Fig. 11, we further explore performance by comparing different topologies with the non-deterministic routing algorithms from Table III under various synthetic traffic patterns with an injection rate of 0.076 packets/cycle. The results are normalized to the latency of the mesh. On average, the dodec reduces latency by 10% compared to the mesh. From the unnormalized data, all networks have the worst performance for bit-complement traffic. However, the example dodec has 50% lower latency compared to mesh. This significant difference is because every packet in bit-complement traffic crosses the network bisection in a mesh/torus network. Torus outperforms mesh for this pattern because it has twice the bisection bandwidth. Randomized node connections enable dodec to balance most regular traffic patterns well; the possibility of forming a hot-spot or hot-link is reduced. Dodecs benefit not only from *shorter diameter* but also from the *randomized node connections* providing better traffic balancing.

E. Full System Simulation

In Fig. 12, we compare the system performance (measured in μops per cycle) for PARSEC [19] normalized to the mesh. We compared the example dodec to other networks with the same per-channel bandwidth (`mesh-eq-ch` and `ring-eq-ch`) and with the same per-router bandwidth (`mesh-eq-rt` and `ring-eq-rt`). With equal per-channel bandwidth, the performance difference between the `mesh-eq-ch` and dodec is negligible. The example dodec outperforms both mesh and ring on all benchmarks when holding per-router bandwidth constant. Compared to `mesh-eq-ch`, dodec achieves competitive performance with a reduction in area and power. Our dodec has the potential to either enable higher application performance or reduce OCN power/area without hurting performance.

We study the variation in per-node average packet latency and report the standard deviation of the average latency across different nodes normalized to `mesh-eq-rt`. Due to the complex and irregular data sharing and resulting traffic patterns, we expect dodec to provide better equality of service [21] due to its randomized connections. Fig. 13 shows that the per-node latency variation in dodec is significantly less than other networks. High per-node latency variation is indicative of unfairness; not all threads receive the same level of service from the network. Furthermore, since dodec enables each thread sees a more uniform memory access

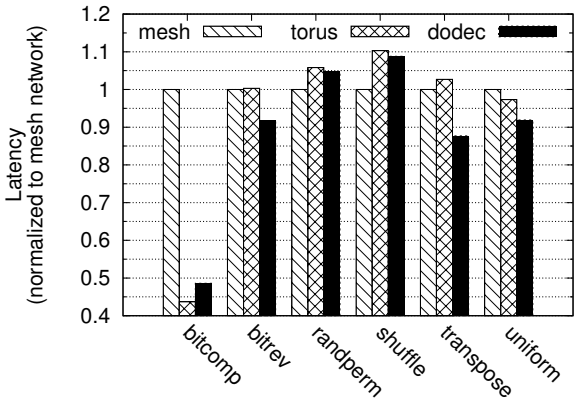


Figure 11: Latency comparison under different traffic patterns

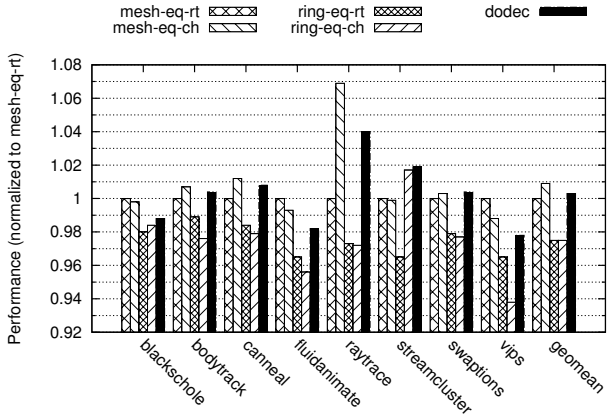


Figure 12: Performance comparison for PARSEC

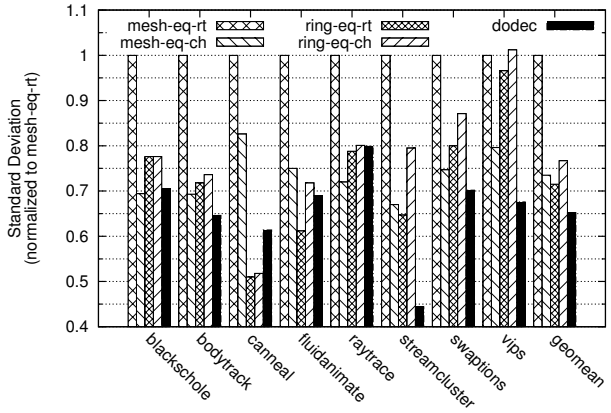


Figure 13: Per-node latency variation for PARSEC

latency, system performance is not adversely affected by thread placement and scheduling. Although random connections can destroy some locality in the network, overall performance improves and nodes receive a more equal level of service. Finally, dodec does not suffer from *congestion* or *hotspots* in the network. In particular, the `mesh-eq-rt` suffers from increased contention due to longer packets and the center nodes in a mesh see a greater level of congestion. As with other topologies, a dodec may not

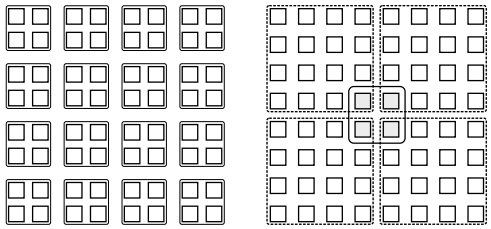


Figure 14: Concentrated (left) & Hierarchical (right) 64-node dodecs

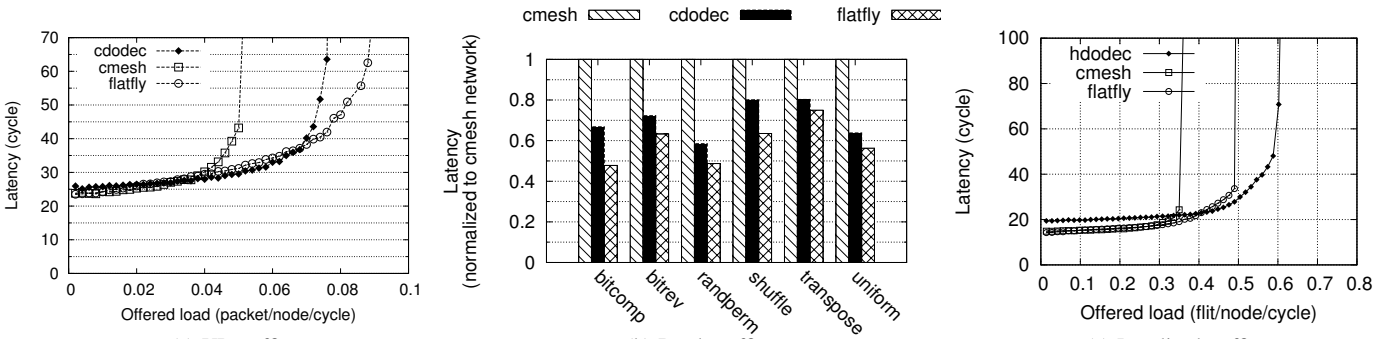
provide the absolute best performance on all traffic patterns; yet, our methodology provides flexibility to customize for particular application scenarios while providing good overall performance in general due to randomized connections and the inherent load balancing they provide.

VI. Discussion

A. Scalability

Network scalability is an important issue in OCNs; however, even design points that do not scale to 1000s of nodes are of interest as much research is needed to make such large networks feasible. The limited number of links per node at radix-3 impacts the scalability of all low-radix networks, including dodec. However, the network diameter of dodecs increases slowly. Typical 36- and 64-node dodecs have diameters of 6 and 8 compared to 10 and 14 for meshes of the same size. The average hop counts are 3.44 and 4.26 for typical 36- and 64-node dodecs compared to 4 and 5.33 for a mesh of the same size. Every topology has a sweet spot in its scalability graph. Dodecs cover a compelling range of scaling points, especially given their amenability to concentration-based approaches. Given their simplicity, they represent a potential middle-ground between small-scale ad hoc networks and large-scale OCNs encompassing perhaps hundreds of nodes. Utilizing the polyhedron OCN methodology with higher radix values improves scalability, load balancing and equality of service for larger networks. Increasing the router radix to 4 results in a random 64-node topology with diameter 3; a radix of 8 yields a diameter of 2.

1) Concentration: Dodec routers have a rich area budget that can be used to expand the channel bandwidth. As average injection rate per node is small in real applications, we can increase channel utilization by having multiple cores share the same channel bandwidth via concentration. We implement a 64-node concentrated dodec (`cdodec`) using a diameter of 3 dodec instance with concentration of 4 (Fig. 14). We compare the `cdodec` with two 64-node topologies: concentrated mesh with express channels (`cmesh`) [2] and flattened butterfly (`flatfly`) [5]. Although `flatfly` achieves a low diameter from its high radix, its performance comes at the price of high-radix routers. Critical router structures such as allocators scale poorly with radix and may require radically different architectures. To demonstrate this issue, we



(a) UR traffic (b) Batch traffic (c) Localized traffic

Figure 15: Results for different 64-node topologies

use a technology-independent delay model [22] to estimate the critical path delay of different routers. We assume that both the dodec and flatfly routers can be implemented as traditional input-queued pipelined routers. The virtual channel allocation stage (VA) typically has the longest delay [1], [23] and determines the maximal router frequency; prior work reports VA delay that is up to 55% higher than switch allocation (SA) [22]. Even with a more modest difference of 5% [23] for a mesh-based router, the delay increases with the number of VCs and radix of the router. We estimate the delay for a separable input-first allocator with matrix arbiters. Using the same configuration, the VA delay in a `flatfly` router is ∼10% higher than the VA delay in a `cdodec` router. This timing advantage directly translates to 10% higher router operating frequency for `cdodec`. An increased critical path may result in additional pipeline stages for high-radix routers.

In Fig. 15a, we compare network performance under uniform traffic. The routing algorithms used for `cmesh` and `flatfly` are O1Turn with express channels [24] and UGAL [5]. We assume the same router microarchitecture as in Sec. V with one more pipeline stage to reflect the increased complexity of many-port routers. We use 4 VCs with 12 flit buffers per VC; deeper VCs cover long link delays. Serialization latency and long-link latency are accurately modeled for all topologies. Due to its short diameter, the `cdodec` has higher saturation throughput than the `cmesh`, which is similar to the comparison between dodec and mesh networks. Compared to `flatfly`, both `cdodec` and `cmesh` have a competitive latency under low to moderate loads. However, the lower diameter in `flatfly` masks the negative effects of long links, and makes its saturation throughput higher than others; many applications are less sensitive to throughput than latency [7].

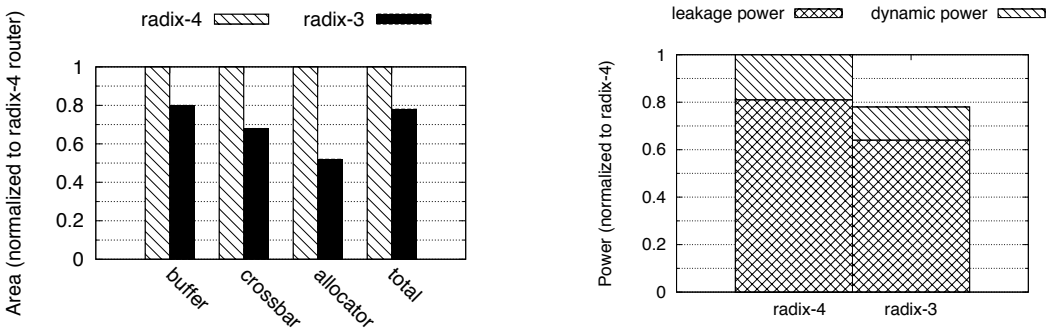
We also perform batch simulations where each node issues a fixed number of read/write requests to the OCN. Each request will trigger a corresponding reply; each node can support 4 outstanding requests. We assume 128-bit messages for read requests and write replies, and 640 bits for read replies and write requests. This leads to 1, 2, and 2 flit short packets for `cdodec`, `cmesh` and `flatfly`. Long packets are 5, 7 and 10 flits. Fig 15b shows the completion time for all requests (1000 transactions/node) for different synthetic traffic patterns. Bursty traffic causes hotspots in the center of the `cmesh` which negatively impacts completion time. Because dodecs have small variation in node latency, we see that `cdodec` better balances traffic, and reduces the system latency dramatically compared to `cmesh`. In addition, with lower implementation cost and complexity, `cdodec` achieves competitive performance compared to `flatfly`.

2) Hierarchy: In large-scale CMPs, it is beneficial to map threads with heavy communication to adjacent nodes to avoid expensive cross-chip communication [25]. Hierarchical OCNs are a good candidate to support applications with high local traffic; dodecs can be scaled hierarchically. We have implemented a 64-node hierarchical dodec (`hdodec`). The network consists of 4 identical 16-node dodec instances (the one used in Sec. VI-A1); the 4 independent dodec sections are the lower-tier of this hierarchical network. In each 16-node section, one node serves as the hub;[8] the 4 hub nodes are fully connected to each other; these nodes (themselves a trivial dodec of size four) form the upper-tier of the network (Fig. 14).[9] `Hdodec` requires a minor modification to routing. Local messages within the same 16-node dodec are routed via the adaptive routing algorithm (Sec. IV). Global messages are first routed to the hub node within the same tier. From there, they are routed to the correct hub node of the destination dodec; from the destination hub, they are then routed to its final destination using the adaptive dodec routing algorithm.

The `hdodec` is optimized for localized traffic; as a result, it does not require the same long links as other 64-node networks such as `flatfly` and `cdodec`. Besides the link length, the router complexity is expected to be less than `flatfly` due to the lower port count. As a result,

[8]We select the corner of the 16-node dodec to minimize the link length between hubs.

[9]One could form a similar hierarchical mesh with 4 16-node meshes and 4 hubs. Comparing this `hmesh` to the `hdodec` produces the same trends as the comparison for the `mesh` and `dodec` in the previous section.



(a) Area (b) Power

Figure 16: Single router comparison in area and power

hdodec could operate at a faster clock frequency compared to flatfly. Using the delay model from Sec. VI-A1, the hdodec router could be 18.5% faster than flatfly with the same canonical router microarchitecture design. In Fig 15c, we compare the 64-node networks under localized traffic.[10] The network setup for flatfly and cmesh is the same as Sec. VI-A1 except that equal per-channel bandwidth is assumed. Hdodec has one less pipeline stage to reflect its advantage of simple router complexity. Hdodec has good saturation throughput in localized traffic; its saturation throughput is almost 70% higher than cmesh and 20% higher than flatfly. However, hdodec has higher latency at low loads due to expensive global messages in hdodec; a message has to detour from the minimal path to route via the hub nodes, which increases the average hop count.

B. Network Cost

As demonstrated, random links provide several benefits for performance in terms of low diameter and improved load balancing. These improvements come with a one-time design cost of higher layout complexity. If we look at the example layout of the on-chip flattened butterfly [5], we see similar layout challenges. By comparison, a Clos network is easily routed [26], yet is significantly more complex than a dodec. Wiring resources for OCNs are fairly abundant which mitigates the added complexity and ensures the routability of our design. For example, consider 9 metal layers available in TSMC $65nm$ technology, $2 \sim 6$ layers might be reserved for local wiring, clock and power; we only need 2 layers to freely route link wires – one for horizontal and one for vertical wires. Previous work [27] uses a $1.2um$ wide bit line ($0.6um$ width with 0.6 spacing) to meet 1GHz timing. Therefore a typical 64-bit wide bidirectional link has a width of $153.6um$. This is negligible compared to the dimension of a typical tile design. For example, a single tile has an area of $2.0mm \times 1.5mm$ area with TSMC $65nm$ [28]. Each tile can accommodate up to 13 parallel 64-bit bidirectional links; yet the most congested tile in our sample dodec has only $4 \sim 5$ links. Therefore, wiring is not a serious problem; we are confident that the dodec does not present a routability problem.

Area and power consumption are two major costs in OCNs. Dodec routers enjoy better area and power efficiency due to their lower radix. Fig. 16 shows area and power results for different radix routers modeled using DSENT [29] in 45nm technology with a 1GHz operating frequency. The area ratio from DSENT matches our synthesis results using open-source OCN router RTL code [30] and Synopsys Design Compiler with a TSMC $65nm$ technology library.

Router area accounts for the majority of the network area; the datapath units (input buffers and switch) dominate router area [2]. Reducing the radix saves both buffer and switch area. Switch size reduction is more significant since it takes more area and decreases dramatically with radix. The area of a crossbar is proportional to the square of the number of router ports. By reducing the number of ports from 5 (mesh/torus) to 4 (dodec), we reduce the switch area by $\sim$36%. According to both DSENT (Fig. 16a) and RTL synthesis, reducing the radix leads to a 20% area savings for a dodec router compared to a mesh/torus router. With this additional area budget, it is possible to increase the channel bandwidth.

Buffers, crossbars and channels consume the majority of OCN power, contributing to $\sim$99% of network power [8]. With equal per-channel bandwidth, the area of these 3 major power contributors can be reduced through radix reduction. Fig. 16b shows a per-router power savings of 22%. Since the leakage power is dominant in submicron processes, the area efficiency of radix-3 router reduces the overall power. Furthermore, as discussed in Sec. III-A, dodec reduces the average hop count compared to a mesh, thus it is expected to further reduce the aggregate per router dynamic power consumption for each packet. In a 16-node network scenario, the estimation on network power reductions show similar trends to router power reductions. Dodec incurs slightly higher link power due to longer wires but still reduces network power by 15%.

Link power can grow significantly in larger-scale networks as link length increases. In Table IV, we compare the required link power calculated by DSENT for cdodec, cmesh and flatfly as described in Sec. VI-A. The table also

[10] We assume intra-section traffic constitutes 80% of the total traffic; destination selection is uniform random for both intra- and inter-section traffic.

	cmesh	cdodec	flatfly
Link Distribution (# short, # medium, # long)	(24, 8, 0)	(7, 10, 7)	(24, 16, 8)
Cost-Driven: Equal Per-Router Bandwidth Design			
Channel Width (bit)	96	128	64
Total Link Power (mW)	160	257	214
Link Power (mW)	(96, 64, 0)	(38, 107, 113)	(65, 85, 64)
Performance-Driven: Equal Per-Channel Bandwidth Design			
Channel Width (bit)	128	128	128
Total Link Power (mW)	215	257	429
Link Power (mW)	(130, 85, 0)	(38, 107, 113)	(130, 170, 129)

Table IV: Link length and power comparison

gives the number of links that span 2 tiles (*short*), 4 tiles (*medium*), 6 tiles (*long*) as well as the total contribution to power consumption of each link type. Repeated links with a 50% toggling rate are assumed. Wire length and channel width are accurately modeled; link power is proportional to both width and length [2]. To achieve its low diameter, `flatfly` requires long links in addition to the cost of complex high-radix routers. In a performance-driven design scenario (equal per-channel bandwidth), `flatfly` suffers from almost twice the link power compared to `cdodec`. This might hinder `flatfly`'s performance because a limited power budget would prevent the network from operating at peak performance. Considering tighter implementation costs (equal per-router bandwidth), we see that `cdodec` can provide high bandwidth with comparable link power. It is higher than the comparison points but further optimization is possible and it is still a worthwhile point in the design space. Link level optimizations such as low-swing signalling [31] can be applied to all networks including dodec.

C. Memory Controller Placement

Memory bandwidth limitations continue to be a problem as CMPs scale [32]. Incorporating multiple memory controllers on a single die increases memory bandwidth. The placement of memory controllers within the OCN can have a significant impact on the system performance as the traffic from/to memory controllers can cause network contention [32]. Randomization in dodecs makes their performance tolerant to these types of hotspots. The regularity of a mesh leads to non-uniform latency and network hotspots; dodecs on the other hand have a more irregular structure which actually leads to lower average latency.

To explore this idea, we compare the impact of different memory controller placements in a 16-node dodec and mesh. These systems have 12 processing elements (PEs) and 4 memory controllers. For mesh, we select the best memory controller placement in prior work [32] and a bad placement with all memory controllers in one corner of the OCN. We calculate *maximal channel load* to score different memory controller placements in dodec and quickly identify the best and worst placements. We use close-loop batch experiments similar to the one described in Sec. VI-A1. In Fig. 17, we show the distribution of completion time for the PEs in each

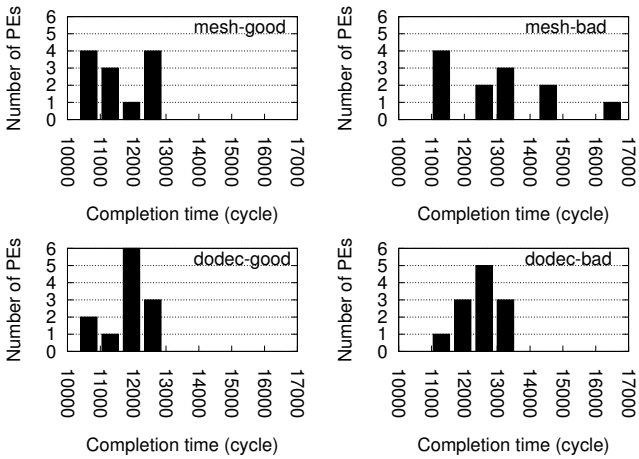


Figure 17: Completion times for different MC placements

scenario. Compared to mesh, dodec enables a lower average completion time with less variation. System performance is less sensitive to the placement of the memory controllers in dodec; this is because that the randomness in dodec provides greater uniformity across nodes. The good dodec completes in 12374 cycles while the bad dodec completes in 13022 cycles; the good mesh completed in 12297 cycles and the bad mesh completes in 16231. We see a much larger variation for the 2 mesh configurations; dodec performance is resilient to the location of hot-nodes within the network.

D. Customizing for Heterogeneity

The irregular node connections in the dodec give a designer considerable room for customized system design. This feature is useful for heterogeneous systems-on-chip (SoCs). If the sizes of different components vary dramatically, fitting all components within a regular topology can be difficult. The dodec topology can overcome this problem because its design has already been optimized for handling irregular topologies. By replacing a traditional regular low-radix OCN, such as mesh, with a dodec, an SoC designer can attain better communication performance while reducing the custom design complexity. In addition, both the network generation process and the corresponding routing computation presented in previous sections are generic and applicable to networks with radices other than 3. Our network design framework also works for networks with heterogeneous node radices. As a result, our methodology can be applied to higher radix routers to improve scalability. The possibility of using different radix routers further extends the design space of communication networks in heterogeneous SoCs. Quantifying the benefits of building a network out of heterogeneous-radix routers requires further study.

VII. Related Work

OCNs commonly use regular, low-radix networks such as meshes [3], [4] and rings [33], [34] due to their low design complexity. Spidergon [35] augments a ring with

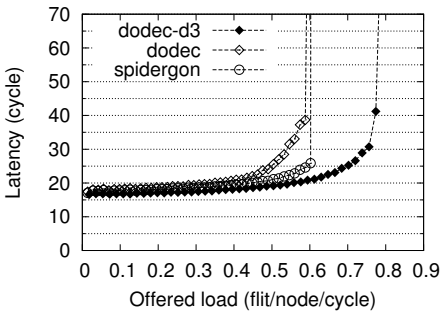


Figure 18: Latency comparison between spidergon and dodec

extra links in an application-specific fashion. We compare our more general dodec approach to a 16-node spidergon with across-first routing [35]. We apply our dodec framework to the same connectivity graph as the spidergon; the resulting dodec has a diameter of 4. We compare the performance of these 2 networks and a diameter-3 dodec (`dodec-d3`) in Fig. 18. All networks use the parameters from Sec. V with equal per-channel bandwidth. The dodec framework is more flexible than spidergon. While the spidergon framework is strictly tied to its fixed topology, our framework can be easily applied to spidergon and achieve competitive performance. In addition, by relaxing the strict cross-connection structure of the spidergon, many possible dodecs emerge. Thus, we provide a much richer design space.

Cube-connected cycles (CCC) [36] uniformly have a node degree of 3. Our work differs from CCCs in 2 major ways. First, CCCs enforce a regular, hierarchical structure to optimize communication locality within a ring, while we embrace a range of networks that share key characteristics. Second, CCCs severely restrict the number of nodes and their connectivity, which limits their applicability as OCNs. A CCC of order n contains $n2^n$ nodes; common OCN sizes such as 16 and 36 cannot be constructed with CCCs. Both spidergon and CCCs are subsets of the general dodec class; however, we make further contributions through our methodology and framework. Compared to spidergon and CCC, we take a more comprehensive look at the entire domain of radix-3 NoCs.

Prior work explores long links in the context of 3D stacked architectures [37] using a clique network. Their network has a diameter of 3: 2 vertical (between stacked dies) hops and 1 horizontal (intra-die) hop. They require higher-radix routers than dodecs and multiple stacked dies, while our low diameter comes at lower cost. Express cubes augment a mesh or torus with long links to bypass the routing latency of one or more nodes [38]. MECS [39] also increases node reach with low cost; MECS uses one-to-many links to increase the inter-node connectivity and reduce diameter in an express cube. Inspired by "small world" networks common in social graphs, additional long links can be inserted in a mesh [40]. All these topologies need additional router ports. Alternatively, we ask: Can we match the performance of a mesh by having fewer, but longer connections?

Random link insertion has been considered for large-scale HPC networks [41]; however, there are key differences between our work and theirs. First, they start with a base topology and add extra random links to reduce the diameter. We build a random-link OCN from the ground up which leads to a high-quality solution. Second, this prior work does not consider link length when inserting extra links. We propose a crucial optimization step to mitigate the impact of long links. Both works note the key benefits that come from randomization. Exploring the trade-offs in an on-chip space is important and is what our polyhedron methodology offers.

VIII. Conclusion

For different applications, there exist topologies that are better from a cost/performance perspective than the ones common today. OCN performance can be improved by increasing node reachability with long, random links. We develop a generic network generation and adaptive routing framework to build different polyhedron OCNs. We explore *dodecs*, an exemplary implementation of network randomization to reduce diameter while simultaneously reducing router radix (and consequently, network cost). We show that the characteristics of a dodec network hold across a distribution of many instances. Under an equal-cost model, we show that diameter is the most significant determinant of performance for a dodec. A 3-radix dodec router saves $\sim$20% area compared to a mesh router. The dodec has lower average network latency and its randomized connections balance traffic load well. Compared to a mesh, a dodec example increases throughput by up to 50% and reduces latency by an average of 10% across a range of traffic patterns at low injection rates. Our polyhedron network methodology opens opportunities for new irregular OCN topologies.

Acknowledgment

The authors would like to thank the anonymous reviewers for their detailed feedback on this work and members of the Enright Jerger research group for their suggestions. We thank Peter Klausler for positing the idea of a dodecahedron-shaped on-chip network. This work was funded by the Natural Sciences and Engineering Research Council of Canada, the Connaught Fund, the Canadian Foundation for Innovation and the University of Toronto.

References

[1] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., 2003.

[2] J. Balfour and W. J. Dally, "Design tradeoffs for tiled CMP on-chip networks," in *Int'l Conf on Supercomputing*, 2006, pp. 187–198.

[3] J. Howard *et al.*, "A 48-core IA-32 processor in 45nm CMOS using on-die message-passing and DVFS for performance and power scaling," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, January 2011.

[4] D. Wentzlaff *et al.*, "On-Chip Interconnection Architecture of the Tile Processor," *IEEE Micro*, vol. 27, no. 5, pp. 15–31, 2007.

[5] J. Kim, J. Balfour, and W. Dally, "Flattened Butterfly Topology for On-Chip Networks," in *Int'l Symposium on Microarchitecture*, 2007, pp. 172–182.

[6] L. Valiant and G. J. Brebner, "Universal schemes for parallel communication," in *ACM Symposium on Theory of Computing*, May 1981, pp. 263–277.

[7] N. Barrow-Williams, C. Fensch, and S. Moore, "A communication characterization of SPLASH-2 and PARSEC," in *IISWC*, 2009.

[8] H. Wang, L.-S. Peh, and S. Malik, "Power-driven design of router microarchitectures in on-chip networks," in *Int'l Symposium on Microarchitecture*, dec. 2003, pp. 105–116.

[9] E. Falkenauer, *Genetic Algorithms and Grouping Problems*. John Wiley & Sons Ltd, 1997.

[10] J. Kim *et al.*, "Microarchitecture of a high-radix router," in *International Symposium on Computer Architecture*, 2005.

[11] W. Dally and C. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Trans. on Comput*, vol. 36, no. 5, pp. 547–553, may 1987.

[12] M. Kinsy *et al.*, "Application-aware deadlock-free oblivious routing," in *International Symposium on Computer Architecture*, 2009.

[13] J. Duato, "A new theory of deadlock-free adaptive routing in wormhole networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 4, no. 12, pp. 1320–1331, dec 1993.

[14] ——, "A necessary and sufficient condition for deadlock-free adaptive routing in wormhole networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 6, no. 10, pp. 1055–1067, oct 1995.

[15] F. Silla and J. Duato, "Improving the efficiency of adaptive routing in networks with irregular topology," in *Int'l Conf on High-Performance Computing*, dec 1997, pp. 330–335.

[16] T. Nesson and S. L. Johnsson, "ROMM routing on mesh and torus networks," in *SPAA*, 1995, pp. 275–287.

[17] L. Gravano *et al.*, "Adaptive deadlock- and livelock-free routing with all minimal paths in torus networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 5, no. 12, pp. 1233–1251, dec 1994.

[18] N. Jiang *et al.*, "A detailed and flexible cycle-accurate network-on-chip simulator," in *ISPASS*, 2013, pp. 86–96.

[19] C. Bienia *et al.*, "The PARSEC benchmark suite: characterization and architectural implications," in *Int'l Conference on Parallel Architecture and Compilation Techniques*, 2008, pp. 72–81.

[20] N. Neelakantam *et al.*, "FeS2: A Full-system Execution-driven Simulator for x86," Poster presented at ASPLOS 2008, 2008.

[21] M. M. Lee *et al.*, "Probabilistic distance-based arbitration: Providing equality of service for many-core CMPs," in *Int'l Symposium on Microarchitecture*, 2010.

[22] L.-S. Peh and W. Dally, "A delay model and speculative architecture for pipelined routers," in *Int'l Symposium on High Performance Computer Architecture*, 2001.

[23] A. Mishra *et al.*, "A case for dynamic frequency tuning in on-chip networks," in *MICRO*, 2009.

[24] D. Seo *et al.*, "Near-optimal worst-case throughput routing for two-dimensional mesh networks," in *International Symposium on Computer Architecture*, 2005, pp. 432–443.

[25] M. R. Marty and M. D. Hill, "Virtual hierarchies to support server consolidation," in *ISCA*, 2007.

[26] Y.-H. Kao *et al.*, "Design of high-radix Clos network-on-chip," in *Int'l Symposium on Networks-on-Chip*, 2010, pp. 181–188.

[27] M. S. Abdelfattah and V. Betz, "The power of communication: Energy-efficient NOCS for FPGAS," in *Int'l Conference on Field Programmable Logic and Applications (FPL)*. IEEE, 2013, pp. 1–8.

[28] S. Vangal *et al.*, "An 80-tile 1.28 TFLOPS network-on-chip in 65nm CMOS," in *International Solid-State Circuits Conference*. IEEE, 2007, pp. 98–589.

[29] C. Sun *et al.*, "DSENT - a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling," in *Int'l Symposium on Networks on Chip*, may 2012, pp. 201–210.

[30] D. Becker, "Efficient microarchitecture for network-on-chip routers," Ph.D. dissertation, Stanford University, 2012.

[31] S. Park *et al.*, "40.4fJ/bit/mm Low-Swing On-Chip Signaling with Self-Resetting Logic Repeaters Embedded within a Mesh NoC in 45nm SOI CMOS," in *Proc. of Design, Automation and Test in Europe*, 2013.

[32] D. Abts *et al.*, "Achieving predictable performance through better memory controller placement in many-core CMPs," in *International Symposium on Computer Architecture*, 2009, pp. 451–461.

[33] S. Bourduas and Z. Zilic, "Modeling and evaluation of ring-based interconnects for network-on-chip," *J. Syst. Archit.*, vol. 57, no. 1, pp. 39–60, Jan. 2011.

[34] J. Kim and H. Kim, "Router microarchitecture and scalability of ring topology in on-chip networks," in *Workshop on Network on Chip Architectures*, 2009, pp. 5–10.

[35] M. Coppola *et al.*, "Spidergon: a novel on-chip communication network," in *Int'l Symp. on System-on-Chip*, Nov. 2004, p. 15.

[36] P. Mazumder, "Evaluation of on-chip static interconnection networks," *IEEE Transactions on Computers*, vol. 36, no. 3, pp. 365–369, 1987.

[37] Y. Xu *et al.*, "A low-radix and low-diameter 3D interconnection network design," in *Int'l Symposium on High Performance Computer Architecture*, 2009.

[38] W. J. Dally, "Express cubes: improving the performance of k-ary n-cube interconnection networks," *IEEE Trans. Comput.*, vol. 40, no. 9, pp. 1016–1023, Sept 1991.

[39] B. Grot *et al.*, "Express cube topologies for on-chip interconnects," in *Int'l Symposium on High Performance Computer Architecture*, 2009, pp. 163–174.

[40] U. 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*, vol. 14, no. 7, July 2006.

[41] M. Koibuchi *et al.*, "A case for random shortcut topologies for HPC interconnects," in *International Symposium on Computer Architecture*, 2012.