

aEqualized: a Novel Routing Algorithm For The Spidergon Network On Chip

Nicola Concer, Salvatore Iamundo, Luciano Bononi
Dipartimento di Scienze dell'Informazione, Università di Bologna
Email: {concer, iamundo, bononi}@cs.unibo.it

Abstract—We present the aEqualized routing algorithm: a novel algorithm for the Spidergon Network on Chip. aEqualized combines the well known aFirst and aLast algorithms proposed in literature obtaining an optimized use of the channels of the network. This optimization allows to reduce the number of channels actually implemented on the chip while maintaining similar performances achieved by the two basic algorithms. In the second part of this paper, we propose a variation on the Spidergon's router architecture that enhances the performance of the network especially when the aEqualized routing algorithm is adopted.

I. INTRODUCTION

A new generation of communication infrastructures called Networks on Chip (NoCs) is considered as a possible alternative to existing On Chip Communication Architectures (OCCAs) based on shared communication medium like on-chip buses [1].

Networks on Chip [2], [3] can be considered in between the classical networking solutions (good for scalability and flexibility to adapt to general communication patterns) and more specific communication and switching architectures for high performance parallel computing (good for performances but less scalable and less flexible under the dynamic configuration viewpoint).

A NoC is composed by three main components: network interfaces, routers and links. Network interfaces (NIs) hide the details of the underlying network providing a standard interface such as Open Core Protocol (OCP). This allows the reuse of IP modules that is a fundamental aspect for multi-core systems on chip (SoC) design. SoC in fact are often composed by heterogeneous and third-party components connected together to form the desired system. Finally NIs provide the packetization and depacketization of the IPs messages so that they can be handled by the underlying network.

Routers can be connected forming the network topology. They implement routing and flow control algorithms and provide buffers used to store the received packets. Routing algorithms compute the next hop for each incoming packet usually implementing simple minimal-path functions.

A NoC topology can be either regular as Mesh, Torus, Tree and others [4], [5] or specifically designed for the traffic it has to support [6], [7], [8]. In the case of regular topologies, routers can implement topology-specific routing algorithms while in customized NoC, routers usually refer to a look up table to compute the next hop of each received packet.

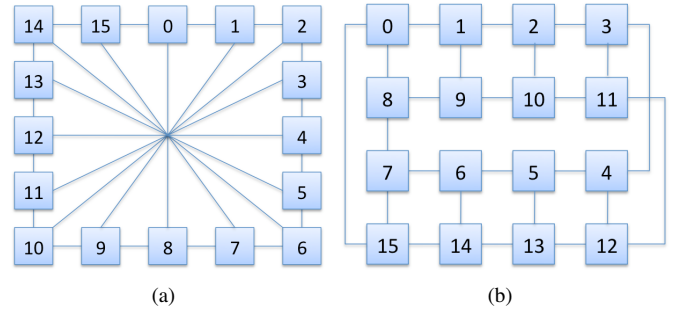


Fig. 1. The Spidergon Architecture (a) and its physical layout (b).

In packet-based NoC communication each packet is split into data units called *flits*. The buffers for channels are defined as multiples of the flit data unit. The packet forwarding among nodes is performed with a flit-by-flit (adaptive, source, arithmetic or table-driven) routing and local signal-based flow control.

Generally the most adopted switching scheme is the wormhole [5] scheme. In wormhole the head flit of a packet is actively routed towards its destination by following the forwarding indications computed by the routers. Subsequent flits are passively switched by pre-configured switching functions on the output buffer of the channel belonging to the path opened by the head flit. When the channel buffer space is available on the input buffer of the channel towards the next switch in the path the next flit of a packet is forwarded on. Flit-based wormhole is an interesting solution compared to virtual cut-through and packet-based circuit switching: its pipelined nature facilitates flow control and end-to-end performances, at the price of allowing packets to be stored in multiple router's buffers hence increasing the probability of channel contentions.

Because of the distributed nature of buffers and channel resources, Networks on Chip can originate circular waiting conditions that can degenerate in deadlock situations [5]. This problem is generally addressed by introducing some restrictions on the routing algorithm or by the adoption of new physical resources called *Virtual Channels* [3] (VC). VCs are implemented by multiple input and output buffers for each physical link. A VC selection algorithm implemented by the router logic, selects the buffer to adopt in order to break the circular dependencies among the channels and solve

the deadlock issue. Moreover VCs can also improve the performance of the system by increasing the throughput on each link [9].

A. Our Contribution

In this paper we propose the *Across Equalized* (aEqualized) algorithm: a routing algorithm that optimally exploits the resources of the system reducing the area and power demands of the NoC.

Our algorithm is thought for networks where a small number of nodes attract most of the network traffic behaving as hotspots. In the multi-core, shared-memory SoC domain this prerequisite is a common situation: a number of Processing Elements (PEs) are often connected to a small number of Storage Elements (SEs) forming a tree of connections with the SEs as roots [10], [11], [12].

II. THE SPIDERGON NOC

The Spidergon [10], [13] architecture represented in Figure 1 is a proposal of ST Microelectronics for the System on Chip domain. It is composed of an even number of nodes (N) and it is similar to a ring enriched by across links between opposite nodes.

Some of the most interesting characteristics of the Spidergon's topology are: *i*) network with regular topology, *ii*) vertex symmetry (same topology appears from any node), *iii*) edge-transitivity, *iv*) constant node degree (equal to 3) translating in simple router hardware and efficiency. High node degree reduces the average path length but increases complexity.

To the best of our knowledge the proposed algorithms for this architecture are the so called *Across First* (aFirst) and *Across Last* (aLast) algorithms [10]. Both algorithms are minimal source routing whose behavior is depicted in Figure 2.

In the aFirst algorithm the across channel can be used only as a first step. Figure 2 shows a Spidergon NoC with node 0 as hot-spot. The arrows starting from each node indicate the path that packets would follow if they were generated by that node. If the destination node is on the source's half of the ring composing the Spidergon NoC, then the packet is forwarded along the clockwise (CW) or counter clockwise (CCW) direction accordingly to the minimal path restriction. If the packet's destination is on the opposite side of the ring the packet is first forwarded along the across channel and from there, along the ring towards its final destination.

In aLast algorithm depicted in Figure 2(b) a packet can traverse an across link only as last step. When a destination is on the source's opposite half of the network the packet is forwarded towards the node connected to the destination through the across link and from there finally delivered. Otherwise, as in aFirst the packet is forwarded along the ring channels in the clockwise or counter clockwise directions.

The restrictions imposed by the aFirst and aLast algorithms are introduced to avoid circular dependencies between the across channels and the ring ones avoiding possible situations of deadlock [5]. Furthermore circular dependencies along the ring channels are addressed adopting two Virtual Channels

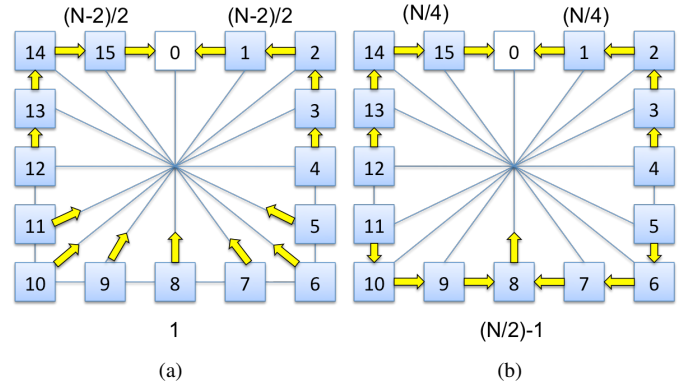


Fig. 2. Nodes' routing behavior towards node 0 using aFirst (a) and aLast (b).

(VC) on each link of the ring. VCs are implemented by multiple input and output buffers for each physical link used to break the circular dependencies formed by the flows of data circulating on the network. Literature proposes a number of VC selection algorithms [5], [3]. In this work we adopted the *multiple dateline VC selection algorithm* described in [9].

III. ROUTING ALGORITHMS CHARACTERIZATION

The routing algorithms described so far are simple to implement and can actually solve the routing issue. A major drawback of these two algorithms is that the across channels are under-utilized as they can be used only once and only as first or last step.

Figure 2 considers a single SE-hotspot (node 0) towards which each PE communicates. In the aFirst (a) case, PE's packets travel essentially through the ring channels so that SE's ring channels must support the traffic generated by $\frac{(N-2)}{2}$ PEs. The SE's across input channel instead is used only by the node set in front of the target (node 8 of Figure 2). On the other hand, the reply packets traveling from SE to the PE nodes on the opposite side of the ring can only use the hotspot's across channel to reach their destination.

In the case of the aLast algorithm (Figure 2(b)) the SE's across channel is over-loaded as it receives the packets generated by the $\frac{(N-1)}{2}$ nodes on the opposite half of the NoC. Channels on the ring instead receive the traffic of only $\frac{(N)}{4}$ nodes each. SE-generated packets instead pass through the ring channels and they may use the across link as last step to get to their final destination.

IV. AEQUALIZED ALGORITHM

The aEqualized algorithm requires the traffic pattern of the NoC to be known at design time and uses this knowledge to redistribute the load of the NoC channels so that all resources are better utilized.

This algorithm combines the aFirst and aLast algorithms seen in Section II. More specifically an aEqualized network is composed by routers implementing either aFirst (tagged as "aFirst") or aLast (tagged as "aLast") routing algorithm.

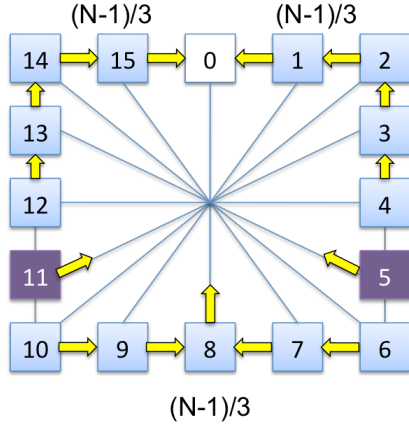


Fig. 3. The Routing aEqualized data-flow considering a single hot spot. Nodes in cyan are tagged as aLast and those in magenta are tagged as aFirst.

The choice of the tag to use depends on the adopted traffic pattern and the position of the hotspot nodes on the network. Restricting the use of only one of the two algorithms per node allows us to avoid additional channel dependencies caused by the across channels and hence new potential deadlock situations.

To explain the tagging operation we start from the simple example shown in Figure 3, that assumes a single hotspot placed on node 0. In Figure 3 nodes in cyan (bright) are tagged as aLast, nodes in magenta (dark) are tagged as aFirst and those in white are the hotspots.

For sake of clarity in Figure 3 and in the following, only $\lambda - \frac{N}{4}$ nodes are actually tagged as aFirst and hence colored in magenta. These nodes are those whose behavior is actually changed by the aEqualized algorithm. In fact, because of their distance from the hotspot, the other nodes will communicate with the SE through the Clockwise or Counter Clockwise link independently from the implemented algorithm.

The node's tagging operation is made by the following steps:

- initially set all nodes as "aLast" so that the hotspot node statistically receives data originated from:
 - $(N/2) - 1$ nodes on the across channel;
 - $(N/4)$ nodes on each ring channel;
- define $M = (N - 1)$ and $R = (M \bmod 3)$
- in function of the remainder R tag the ϕ nodes on the clockwise and counter clockwise side of the hotspot node as aFirst and the remaining λ nodes as aLast in such a way that:

$$\begin{aligned} R = 0: & \phi = M/3 \text{ and } \lambda = M/3; \\ R = 1: & \phi = M/3 \text{ and } \lambda = M/3 + 1; \\ R = 2: & \phi = M/3 + 1 \text{ and } \lambda = M/3; \end{aligned}$$

- tag all the Storage Elements as "aFirst" (the reason will be clear later)
- note that $2 * \phi + \lambda + 1 = N$

By tagging the nodes in the proposed way we define three sets of nodes each one communicating with the hotspot node

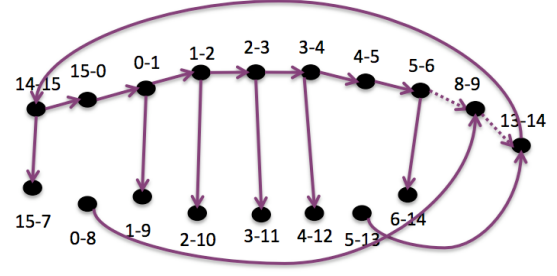


Fig. 4. Channel Dependency Graph of the aEqualized algorithm.

through a different SE's input channel. The three sets have a size that differs at maximum of one node depending on the network size N .

Figure 4 depicts the *Channel Dependency Graph* (CDG) [5] of the network in Figure 3 without considering the VCs on the ring channels. In a CDG, nodes represent channels of the network and edges represent the possible dependencies generated by packets traversing them. For sake of clarity we report a fraction of the complete CDG where we consider only the dependencies of the clockwise channels on the Spidergon's ring plus the related across channels. The CDG for counter-clockwise connections is similar. Figure 3 shows that aLast across links can be used only as last step, hence their relative nodes in the CDG have no outgoing arrows. Instead aFirst channels can be used only as first step so that in the CDG they have no incoming edges. Alternating aFirst and aLast routers then does not generate any dangerous dependency. In fact only the links on the ring generate a cycle of dependencies and as in the two basic algorithms, this must be addressed through VCs.

Figure 5 depicts a complex example where more SEs are inserted. Here nodes are tagged accordingly to the position of the three hotspots. Note that now nodes may have up to three outgoing arrows indicating the different paths followed by the generated packets when directed to one of the three possible destinations. It's clear that by considering a uniform traffic directed to the SE nodes, when the network size is not multiple of three the traffic equalization will result less balanced.

As a consequence of the tagging restrictions imposed by the aEqualized algorithm, request and reply packets exchanged between any PE - SE pair follow the same path but in a opposite direction. This characteristic allows to better exploit the channels of the network and also allows to reduce the number of across links actually used by the packets traversing the Spidergon NoC.

A main drawback for this algorithm is that it imposes some restrictions on the mapping of the devices on the network: aFirst nodes should not be used as last step to reach a destination through the across channel as the router won't have the right to use the requested link. Hence a storage element should not be placed on a node connected to an aFirst-tagged router by the across link. Figure 6 shows an example: two SE-hotspot nodes (nodes 0 and 8) are placed on the same across

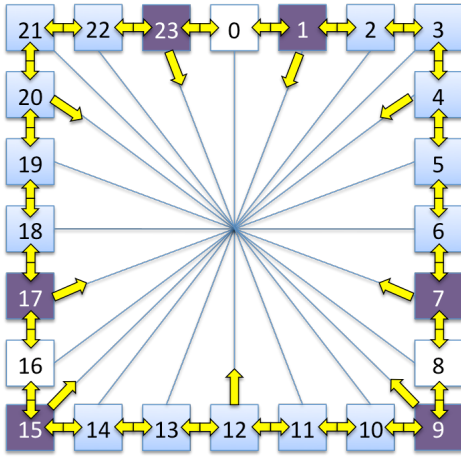


Fig. 5. The Routing aEqualized data-flow considering three hotspots.

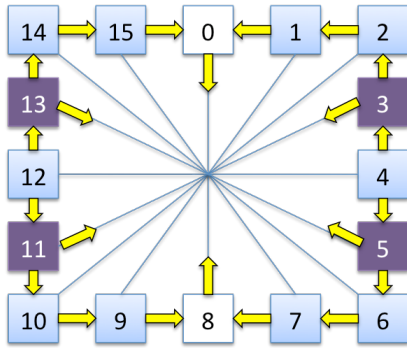


Fig. 6. A drawback of the aEqualized routing algorithm: hotspot should not be connected to aFirst-tagged nodes.

channel and both hotspots are tagged as “aFirst”.

Let’s consider a communication between node 10 and node 0. Node 10 is tagged as “aLast” so it would forward its packets in the counter clockwise direction towards node 8 that is supposed to forward them to node 0. Node 8 though is tagged as “aFirst” so it does not have the right to use the across channel as last step. Instead, it would forward the packets also in the counter clockwise direction towards node 7 breaking, in this way, the minimal path restriction.

V. SYSTEM LEVEL ANALYSIS

The Spidergon architecture has been modeled and configured with synchronous interconnected routers. Each router was provided with two virtual channels on the clockwise and counter clockwise channels and was linked to a single external IP through a network interface layer as depicted in Figure 7.

Depending on the simulated scenario, each IP acted either as a PE or as SE. PEs (also called initiators) generate request packets directed to the SEs selected in a random way with a uniform probability distribution. SE nodes receive the requests packets from the PE and return response packets to the source IP modeling a *shared memory* system. In our analysis we assume that all PEs/SEs are provided with infinite FIFO output

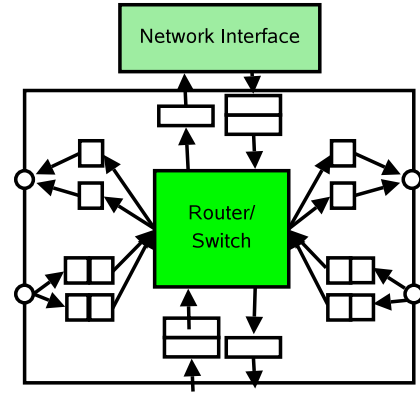


Fig. 7. Standard node router with 2 virtual channels on the CW and CCW (left and right) links and one on the Across and NI ones (up and down).

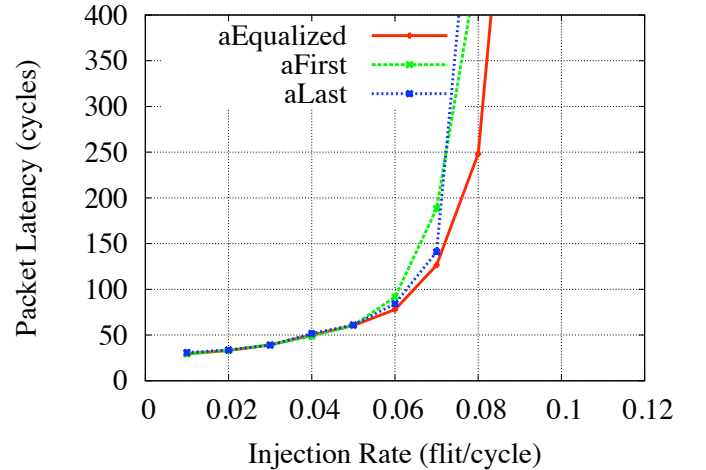


Fig. 8. Average packet latency of a 24-nodes Spidergon with three hotspot Storage Elements.

buffers that temporarily store the produced packets once the underlying NoC is not able to absorb them at a sufficient rate.

We compared the considered scenarios by measuring the packet latency computed as the average time (in clock cycles) taken by a packet to enter the network, traverse it and reach its final destination.

Figure 8 shows the average packet latency on a Spidergon NoC with 24 nodes with three hotspots uniformly addressed by the PEs. The three algorithms perform in a similar way and only aLast performs slightly worse than the others.

The benefits given by the traffic equalization on the hotspots’ channels are hidden by the bottleneck present at the hotspots’ Network Interface channel. Here packets are blocked by the great amount of data contending the same shared resource: the channel towards the NI device.

On the other hand the main improvement introduced by aEqualized algorithm is not on the performance side but rather on the resource requirements side. We consider a physical link as required if there are packets passing through it. In SoC environment traffic patterns are usually fixed and well known so that, in the final physical design, the unused links

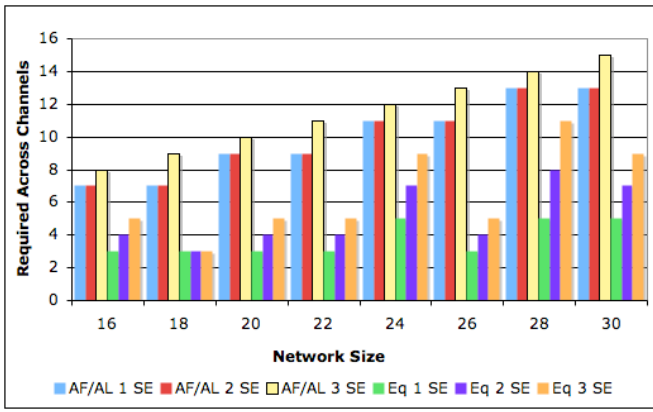


Fig. 9. Number of required Across bidirectional links on a Spidergon NoC with respect to the size of the network.

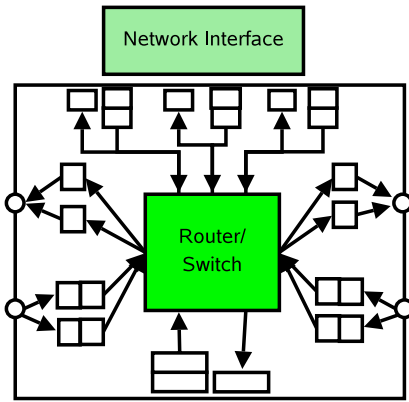


Fig. 10. Spidergon router with three links towards to the Network Interface.

and router ports can be avoided saving costs, area and power consumption.

Considering all the connections as bidirectional we noted that, in all the experiments, all the ring channels have been actually traversed by (request or reply) packets. Instead the number of Across channel actually used depends on the selected routing algorithm and the number of SEs installed on the network.

Figure 9 reports the number of across links (considered bidirectional) actually used by the packets with respect to the size of the network and the number of hotspots in the sistem (i.e. one, two and three). By granting essentially the same performance as aFirst and aLast, aEqualized algorithm uses up to 60% less across channels than the other two algorithms (network with 26 nodes). This improvement is due to the fact that request and reply packets pass through the same path in the opposite direction. AFirst and aLast algorithms instead use all the across channels of the network either to forward request (aFirst) or reply (aLast) packets. By adopting a uniform traffic pattern, considering also the ring channels of the NoC, the aEqualized algorithm allows a reduction of the required bidirectional connections by up to 13% by maintaining similar performances.

A veritable bottleneck in the Spidergon architecture is

the single link connecting the router to the node's Network Interface. The hotspot node is the target collecting a large amount of traffic and its NI channel is the single connection handling all the incoming data. Figure 10 shows a variation on the architecture of the Spidergon node that we investigate to improve the performance of the network. Here a node is provided with three independent physical links each one directed towards the Node's NI. Each NI channel is dedicated to a single and pre-defined input channel in order to remove the contention towards the Network Interface. Packets generated by the NI and directed to the underlying router are placed in one of the three randomly selected NI output buffers.

Figure 11 shows the average packet latency measured with the proposed router improvement on a 24-nodes Spidergon with one, two and three targets and a random uniform traffic pattern.

Once removed the NI bottleneck, aEqualized clearly outperforms the other two routing algorithms, also reducing the required physical links. The worst performing algorithm is aFirst as only the nodes facing the hotspots can use the across channel and the NI links dedicated to it. The rest of the traffic reaches the hotspots through the two ring-related channels. ALast has a variable performance as the hotspots across channels may be overloaded of traffic. AEqualized algorithm instead optimizes the use of all three new NI channels sensibly improving the performance of the system.

Simulations with one, two and three hotspots and different network sizes of 8 up to 30 nodes confirm the reported results.

VI. CONCLUSIONS

We presented a novel routing algorithm called *Across Equalized* or aEqualized for the Spidergon Network on Chip. AEqualized balances the traffic on the input channels of the network's hotspots by assigning to each router either the aLast or aFirst routing algorithm. This "tagging" operation depends on the position of each single node with respect to the hotspots of the network.

Considering the standard implementation of a Spidergon router: with four ports (NI, Across, Clockwise and Counter Clockwise), in our case study, this algorithm essentially maintains the performances of the other two algorithms by reducing the number required across links by up to 60% and the general required connection by a 13%.

Considering an enriched version of the routers with three ports towards the network interface this algorithm sensibly improves the performance of the system by maintaining the reduced amount of links and buffers.

In our future work we plan analyze the aEqualized routing algorithm under more realistic traffic patterns and to improve the enriched routers channels assignment policy.

ACKNOWLEDGMENT

The authors wish to thank Marcello Coppola and Riccardo Locatelli (ST Microelectronics) for their contributions and constructive discussions about the topics and analysis presented in this paper.

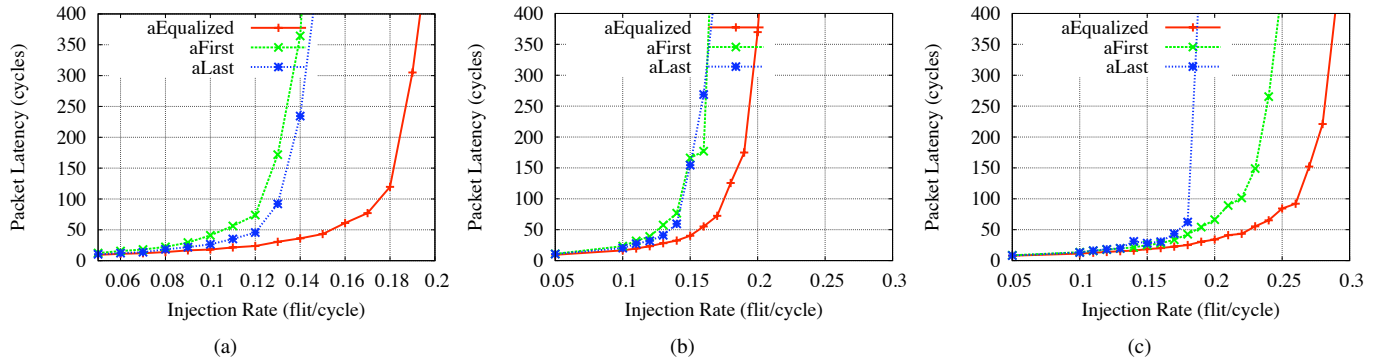


Fig. 11. Average packet latency of a 24-nodes Spidergon with one (a), two (b) and three (c) hotspots using three independent NI channels.

REFERENCES

- [1] G. de Micheli and L. Benini, "Networks on chip: A new paradigm for systems on chip design," in *Proceedings of the conference on Design, automation and test in Europe*, 2002.
- [2] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the Design Automation Conference (DAC) 2001*.
- [3] G. D. Micheli and L. Benini, *Networks on Chips: Technology and Tools (Systems on Silicon)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2006.
- [4] L. Bononi and N. Concer, "Simulation and analysis of network on chip architectures: ring, spidergon and 2d mesh," in *Design, Automation and Test in Europe (DATE'06)*.
- [5] J. Duato, S. Yalamanchili, and N. Lionel, *Interconnection Networks: An Engineering Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002.
- [6] L. Bononi, N. Concer, M. Grammatikakis, M. Coppola, and R. Locatelli, "Noc topologies exploration based on mapping and simulation models," in *Proceedings of the 10th Euromicro Conference on Digital System Design (DSD)'07*.
- [7] S. Murali, P. Meloni, F. Angiolini, D. Atienza, S. Carta, L. Benini, G. D. Micheli, and L. Raffo, "Designing application-specific networks on chips with floorplan information," in *Proceedings of the International Conference on Computer-aided design (ICCAD)'06*.
- [8] K. Srinivasan, K. S. Chatha, and G. Konjevod, "Application specific network-on-chip design with guaranteed quality approximation algorithms," in *Proceedings of the Conference on Asia South Pacific Design Automation (ASP-DAC)'07*.
- [9] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003.
- [10] M. Coppola, M. D. Grammatikakis, R. Locatelli, G. Maruccia, and L. Pieralisi, *Design of Cost-Efficient Interconnect Processing Units: Spidergon STNoC*. Boca Raton, FL, USA: CRC Press, Inc., 2008.
- [11] A. Hansson and K. Goossens, "Trade-offs in the configuration of a network on chip for multiple use-cases," in *Proc. of the Int. Symp. on Networks on Chip (NOCS)*.
- [12] M. Kim, D. Kim, and G. E. Sobelman, "Mpeg-4 performance analysis for CDMA network on chip," in *Proceedings on the International Conference on Communications Circuits and Systems*, 2005.
- [13] M. Moadeli, A. Shahrabi, W. Vanderbauwhede, and M. Ould-Khaoua, "An analytical performance model for the Spidergon noc," in *Proceedings of the 21st Int. Conference on Advanced Networking and Applications (AINA '07)*.