International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 5, No.2 (March-2016)

http://dx.doi.org/10.12785/ijcds/050202

The Performance of NoCs for Very Large Manycore Systems under Locality-based Traffic

Sharifa Al Khanjari[1] and Wim Vanderbauwhede[1]

[1] *School of Computing Science, University of Glasgow, Glasgow, UK*

Received 20 July 2015, Revised 9 December 2015, Accepted 17 January 2016, Published 1 March 2016

Abstract: The scaling of semiconductor technologies is leading to processors with increasing numbers of cores. A key enabler in manycore systems is the use of Networks-on-Chip (NoC) as a global communication mechanism. The adoption of NoCs in manycore systems requires a shift in focus from computation to communication, as communication is fast becoming the dominant factor in processor performance. In large manycore systems, performance is predicated on the locality of communication. In this work, we investigate the performance of three NoC topologies for systems with thousands of processor cores under two types of localised traffic models. We present latency and throughput results comparing fat quadtree, concentrated mesh and mesh topologies under different degrees of localisation. Our results, obtained using a modified version of the HNOCS NoC simulator and based on the ITRS physical data for 2023, show that the type of locality traffic and the degree of localisation significantly affects the NoC performance, and that scale-invariant topologies perform worse than flat topologies.

Keywords: Manycore, NUMA, Network on Chip, Locality

1. INTRODUCTION

With the drive towards exascale computing and the resulting need for reduction in power consumption and optimization of performance per Watt, a growth in the number of cores per chip is to be expected. In addition, if the semiconductor industry can maintain scaling according to Moore's law, then next decade's generation of multiprocessor systems on chip will contain hundreds to thousands of cores. Such a massively manycore system requires high performance interconnections to transfer data between the cores on the chip. For manycore processors with close to 100 cores such as the Tilera Tile64 [1] or the Intel MIC [2], Networks on chip (NoC) have become the preferred on-chip communication infrastructure. Performance of NoC based manycore systems is highly dependent on the traffic patterns and the NoC topologies. In manycore systems, communication, not computation is the performance-limiting factor. In ten years, according to the International Technology Roadmap for Semiconductors (ITRS), we can expect that a thousand-core CPU can fit into an area of less than $1\,mm^2$ and it can consume only a few Watts. A further trend is the 3D-stacked memory [3, 4], already the Xeon Phi uses this technology and integrates it with the CPU using flip-chip. Further integration leading to memory stacked on top of the CPU is in active research.

Consequently, future manycore platforms can reasonably be expected to have such essential distributed memory architecture. Because of the large difference in access time between memory placed on top of a core and memory placed a far removed core, a message passing style of programming similar to the approach used in NUMA architectures and HPC clusters is to be expected. In fact, it is likely that users will want to deploy legacy MPI code on these novel platforms, because rewriting large HPC codebases is a very large effort. However, other message-passing approaches such as Erlang [5] would be equally suitable for this type of architecture. Regardless of the programming language, it is clear that there is a need for programming models that exploit locality to avoid the long latency of communication between remote cores.

The NoC architecture for such manycore CPUs for exascale systems is of particular interest. Besides the standard mesh (as used in e.g. the Tilera manycore system and the Intel SCC) and the ring topology used in the Intel Xeon Phi and the recent Intel Xeons [2], many NoC topologies have been proposed and evaluated. Some researches aimed to provide improvements on the ring topology, such as the Spidergon [6] or our own Quarc [7]; some researches aimed to improve on the mesh, e.g. the concentrated mesh [8]. There have also been some

E-mail: s.al-khanjari.1@research.gla.ac.uk, wim.vanderbauwhede@glasgow.ac.uk

researches interested in scale-invariant topologies such as the quadtree [9].

Recently, locality-based traffic has been receiving increased attention [10,11]. Locality of computation is essential to reduce latency and increase performance. As the number of cores increases, locality will only become more important. Without locality, the amount of communication grows with the square of the number of cores. As a consequence, applications running on high-performance compute clusters always display strong locality. Moreover, it has long been known that for high performance, message-passing applications require locality. We contend that a combination of locality-aware task placement and a locality-based communication topology can greatly improve performance of message-passing style applications.

For this particular work we have taken as a starting point the common patterns used in Computational Fluid Dynamics codes. Most of the run time of e.g. a weather simulation is spent in solving differential equations [12]. These algorithms make extensive using of stencils for neighbour-based interaction, but they also involve whole-system reductions and this reduction step is often what determines the performance. Such reductions are most efficiently done using a tree to compute and aggregate partial results, rather than by a single process. Thus they produce a different type of localised traffic pattern.

In order to investigate the effect of the topology on the performance for similar computational patterns, in this paper we propose abstract models of locality with different distance metrics, which let us control the degree of locality as well as the shape of the local area, and thus provides general insights into the suitability of a given topology for a given locality-based model.

The remainder of this paper is organized as follows. In Section 2.A we present our locality-based models. In Section 2.B we show that our locality-based models are related to Rent's rule. In Section 3 we describe the topologies used to observe how the locality-based models affect their performance. In Section 4 we detail the cost model of the topologies, the technology node assumptions used in this work, the overheads for a 1024 core chip and packet formatting and switching techniques used. In Sections 5 we describe evaluation methodologies and present results and analysis. We conclude in Section 6.

2. NON UNIFORM TRAFFIC SCENARIOS BASED ON LOCALITY OF COMPUTATION

A. Hierarchical Model for Locality-based Traffic

We propose a simple hierarchical model for locality-based traffic. To model locality, we group the cores of the chip and create hierarchical groups to encompass the whole system. Using $0 < l \leq n$ for the levels of the hierarchy, and assuming $n = log_4(N)$ with N the number of cores, we can express the probability for communication across level-l as:

$$p(l) = (l - \alpha)\alpha^{l-1}, 1 < l \leq n \quad (1)$$

$$p(n) = \alpha^{n+1}$$

The parameter α relates to the locality of the processes making up a message-passing based task. It expresses the probability that a message has to travel a certain distance in the hierarchy. Larger α means lower locality: if $\alpha = 0.8$, then according to equation (1) 80% of the messages will have a destination outside the first cluster, and 80% of that portion outside the second cluster, etc. When $\alpha = 1$, it means that most of the traffic will be sent to the last level cluster. This is worse than a uniform traffic where all destinations have equal probability of being the destination.

In this paper, we use two different instances of this locality-based model to evaluate the performance of the NoC topologies.

• In the first model (Group Clustering), we group the cores of the chip per four, and create hierarchical groups to encompass the whole system, in a scale-invariant fashion. In this case $n = log_4(N)$ with N the number of cores and each level contains 4^n cores. This is a generalisation of the reduction traffic.

• In the second model (Ring Clustering), we group the cores of the chip in concentric rings around the sender core. In this case the number of cores per level is $8n$ as long as the rings don't meet the edge. This is a generalisation of the nearest-neighbour (stencil) traffic.

It is interesting to note that our locality-based models are actually instances of the hotspot traffic. Hotspot traffic is when messages are destined to a specific core with a certain probability and are otherwise uniformly distributed. According to the hotspot traffic model each core first generates a random number. If it is less than a predefined threshold, the message is sent to the hotspot core. Otherwise, it is sent to other nodes in the network with a uniform distribution. Our locality-based models perfectly fit this definition.

B. Relationship To Rent's Rule

There is a very interesting relationship between the α in our locality-based model and the measure for locality known as Rent's rule, as extended to NoCs by Greenfield et al. in [13].

Rent's rule was described in 1971 by Landman and Russo as the relation between the number of terminals at the boundaries of an electronic circuit and the number of internal components, such as logic gates [14].

Greenfield et al. [13] argued that network traffic follows Rent's rule. In addition, Rent's rule will naturally arise in multi- and many-core because just as it is usually

undesirable to place two cores on opposite ends of the chip and connect them; it is also undesirable to map two communicating tasks to tiles at opposite ends. They extended the concept of connection locality in circuits to communication locality among cores, proposing a bandwidth based version of Rent's rule,

$$B = kG^{\beta}$$

where B is the bandwidth sent or received by a cluster of G network nodes, k is the average bandwidth per node, and $0 \leq \beta \leq 1$ is the Rent's exponent.

Heirman et al. [15] showed experimentally that many parallel applications follow Rent's rule. They analysed a variety of popular benchmark applications running on 32 and 64 cores network. Using a hierarchically partitioning algorithm, they showed that the programs follow Rent's rule with measured values of the Rent's exponent β ranging from 0.55 to 0.74 which proves that communication is definitely localised.

As our manycore architecture is NoC-based, all traffic is transferred over the NoC. We are then principally concerned with the latency and the throughput of our message passing communications and hence with the latency and bandwidth of our NoC. In our proposed architecture, the traffic flowing from level l to level $l + 1$ depends on α and the number of cores in the level, 4^l. Thus, for each level, the amount of core traffic generated is proportional to the $B_C(l) = 4^l(1 - \alpha)\alpha^{l-1}$. Following the derivation in [13], we consider the ratio of traffic between to subsequent levels:

$$B_C(l + 1)/B_C(l) = 4\alpha$$

and generalised to k levels:

$$B_C(l + k)/B_C(l) = (4\alpha)^K$$

Following again [13], we define $\alpha = 4^{\beta-1}$ and $x = 4^k$ and set $l = 1$, and we obtain exactly the same equation as in [13], expressing Rent's rule for bandwidth.

$$B_C(x) = B_C(1)x^{\beta} \qquad (2)$$

Thus we obtain the very interesting result that the traffic bandwidth following from the distribution in equation (1) is governed by Rent's rule. In other words, our proposed hierarchical approach results in Rent's rule for the bandwidth of the generated traffic.

3. NETWORK TOPOLOGIES

The network topology describes how routers are connected with each other and with the cores. For manycore systems, the communication cost is increasingly important. The topology has a major impact on the scalability and the performance of the network. With this motivation, we analyse the network performance and communication locality in flat and hierarchical topologies. In this section we describe the three different types of network topologies namely the mesh, the concentrated mesh and the fat quadtree.

A. Mesh

The mesh topology has been the most popular NoC topology so far and it has been used in most of the recent manycore chips such as Intel SCC 48-core [16], TFlops 80-core [17], Tilera 64-core [1]. It organises the routers in a grid, one router per core. Addresses of routers and cores can be easily defined as x and y coordinates in mesh. Figure 1 shows the layout for mesh for 64 nodes. The mesh has a radix (number of ports) of 5. Deadlock is avoided by using XY routing which is a deadlock free routing algorithm.

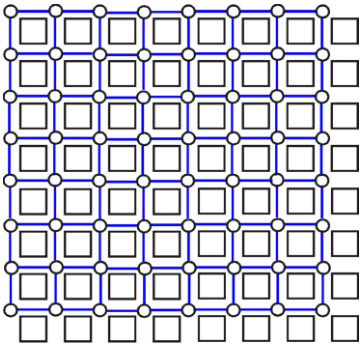

Figure 1. Mesh

B. Concentrated Mesh

The concentrated mesh (cmesh) has been introduced by [8] to preserve the advantages of a mesh with decreased diameter. The number of cores sharing a router is called the concentration degree of the network. In this work we use degree of 4. Figure 2 shows the layout for concentrated mesh for 64 nodes. The concentrated mesh topology requires less number of routers resulting in reduced hop count and consequently improved latency over mesh. It has a radix of 8. The routing is the same as in the normal mesh.

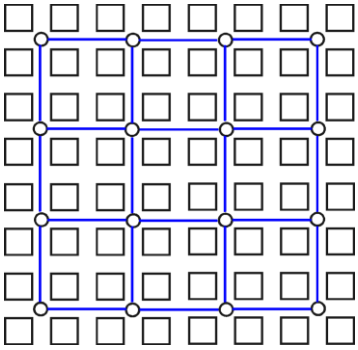

Figure 2. Concentrated Mesh

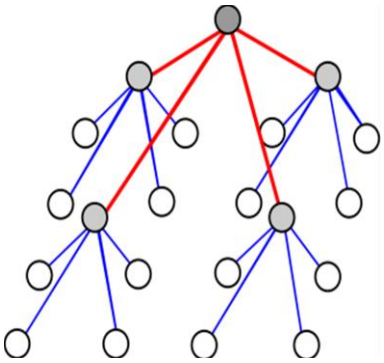

Figure 3. Logical Fat Quadtree Layout

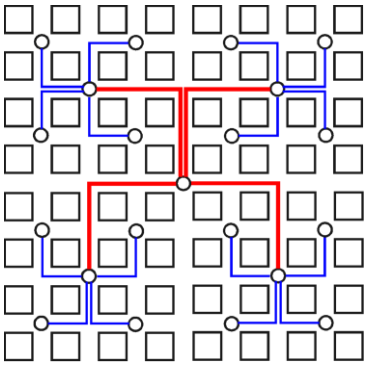

Figure 4. Physical Fat Quadtree layout

C. Fat Quadtree

The fat tree connects routers in a tree with the cores at the leaves. To avoid congestion towards the root of the tree, a fat tree use an increasing number of point-to-point links per connection as described in [9]. The number of links is multiplied by the tree degree (degree 4 in this work) as we move toward the root. A fat quadtree of size N is a structure that can be regarded as a rooted 4-ary tree of height $log_4(N)$. In this way it exactly reflects the group clustering model. Figure 3 shows the logical layout for a fat quadtree and figure 4 shows the physical layout for a fat quadtree for 64 nodes. The fat quadtree has $(4N-1)/3$ routers. The advantage of the fat quadtree over the mesh is that the communication diameter of a fat quadtree is only $O(log_4 N)$ compared to $O(\sqrt{N})$ for the mesh. It uses nearest-common ancestor routing. Packets are adaptively routed up to the common ancestor and deterministically down to the destination. The fat quadtree is deadlock-free.

4. NoC Topologies Overhead

A. Cost Model

We present the cost model of the mesh, the concentrated mesh and the fat quadtree in terms of link complexity, number of routers and buffers. Table 1 shows the notations used for the cost model.

Link Complexity is the total number of links in the topology. Note that the mesh and the concentrated mesh have two virtual channels while the fat quadtree has no virtual channels. In section 4.C, we will compute the wire overhead for mesh, concentrated mesh and fat quadtree. The number of routers in mesh has the order of $O(N)$ compared to $O(N/3)$ in a fat quadtree and $O(N/4)$ in a concentrated mesh.

TABLE I. TABLE OF NOTATIONS

Symbol	Description	Symbol	Description
N	Number cores	N_L	Number of links
n_B	Number of buffers	N_R	Number of routers
n_{VC}	Number of virtual channels	N_B	Number of buffers

The total number of buffers in the mesh and the concentrated mesh are straight forward as in each router there are 5 and 8 ports, respectively. The total number of buffers in the fat quadtree is more complex since the buffer size is doubling at every level because the wire lengths are doubling at every level. Table 2 shows the cost model in terms of link complexity, number of routers and buffers and the values for 1024 cores.

TABLE II. COST MODEL FOR 1024 CORES

Mesh		
N_L	$2\sqrt{N}(\sqrt{N}-1)n_{VC}$	4094
N_R	N	1024
N_B	$5n_B n_{VC} N$	163840
Fat Quadtree		
N_L	$Nlog_4(N)$	5120
N_R	$\frac{N-1}{3}$	341
N_B	$n_B(N-4)(\sqrt{N}-2)+2n_B\sqrt{N}$	490624
Concentrated Mesh		
N_L	$\sqrt{N}(\frac{\sqrt{N}}{2}-1).n_{VC}$	960
N_R	$\frac{N}{4}$	256
N_B	$2n_B n_{VC} N$	65536

To calculate the wire link overhead, we get the links width $\text{Link}_{\text{width}}$ as in equation (3), where (W) is the wire pitch, (N_{bits}) is the number of bits in parallel for one packet and (N_{layers}) is the number of layers.

$$Link_{width} = \frac{W \times N_{bits}}{N_{layers}} \quad (3)$$

The number of vertical wires in the fat quadtree can be obtained using $Vertical_{wire} = 2^{(log4N-1)} log_4 N$, and for the mesh $\text{Vertical}_{\text{wire}} = \sqrt{N}$ where N is the number of cores. Starting from equation (3) and the number of vertical wires, we can compute the area overheads as follows:

$$Area_{Wire} = 2 \times Width_{VerticalLink} \times Width_{Chip} \quad (4)$$

$$Area_{Chip} = Area_{Core} \times N + Area_{Wire} \quad (5)$$

$$Area_{Overhead} = \frac{Area_{Wire}}{Area_{Chip}} \quad (6)$$

B. Technology Node Assumptions

To ensure realistic simulations of the next decade's manycore systems, we assume the 10 nm process in 2023 as projected by the International Technology Roadmap for Semiconductors. Table 3 lists the physical parameters for this technology node from the 2011 ITRS data [18]. We used the die size of the 64-core Tile64 from [19] ($433.5\ mm^2$) to estimate the core size and scaled it to the 2023 node. From ITRS the chip size at production in 2013 was 140 mm^2 so it is less than the estimated core size by a factor of 3.1 $(434/48 = 140)$. In 2023, the chip will contain $20\times$ more cores than today's chip. Consequently, the chip size in 2023 with nearly 1280 (20×64) cores will be approximately $3.1 \times 111 = 344.1\ mm^2$. Hence, the area of one core is $344.1/1280 = 0.3\ mm^2$. This area corresponds to dimensions of about $0.5\ mm \times 0.5\ mm$. The width of one core is thus $0.5\ mm$ and the wire delay can be estimated at $0.5 \times 33827/1000 = 17.5\ ns$.

TABLE III. TECHNOLOGY PARAMETERS FOR 10NM CMOS (2023) BASED ON ITRS 2011

Year	2013	2023
Chip size at production	140 mm^2	111 mm^2
Global wire delay	1 ns/mm	33.8 ns/mm
Estimated core size	6.8 mm^2	0.3 mm^2
Estimated wire delay	2.6 ns	17.5 ns

C. Overheads for a 1024-core chip, 10nm (2023) ITRS node

In terms of buffer space overhead, the mesh will require 5.1 KB of storage per core, the concentrated mesh will require 2 KB while the fat quadtree will require 15.3 KB. Although the total number of buffers is a lot more in fat quadtree than mesh, it is only a very small fraction to the total size of the chip, e.g. just the per-core L2 cache on the 60-core Xeon Phi is already 512 KB. With the above assumptions, the area of a 15.3 KB SRAM buffer would be 0.14% of the estimated core size (memory density 37.6 MB/mm^2).

In terms of wire overhead, our cost model shows that the wire area overhead for the fat quadtree would be 0.3% of the estimated chip size for a 1024-core chip (wire pitch 17 nm).

These results are very important as they indicate that for this type of manycore architecture, the NoC overhead is negligible, which means that the choice of the NoC can be based solely on performance.

D. Packet Format and Switching

In this work, we use an edge case of wormhole switching which uses only a single flit, and is consequently identical to virtual cut-through and store-and-forward, but with a single-flit. This has an advantage over the wormhole switching with more flits per packet because it will not block the routers along the packet path. Lee et al. [20] argued that widening the flit size will increase the network on chip performance and it is cost effective. In addition, in a modern chip wires are cheap.

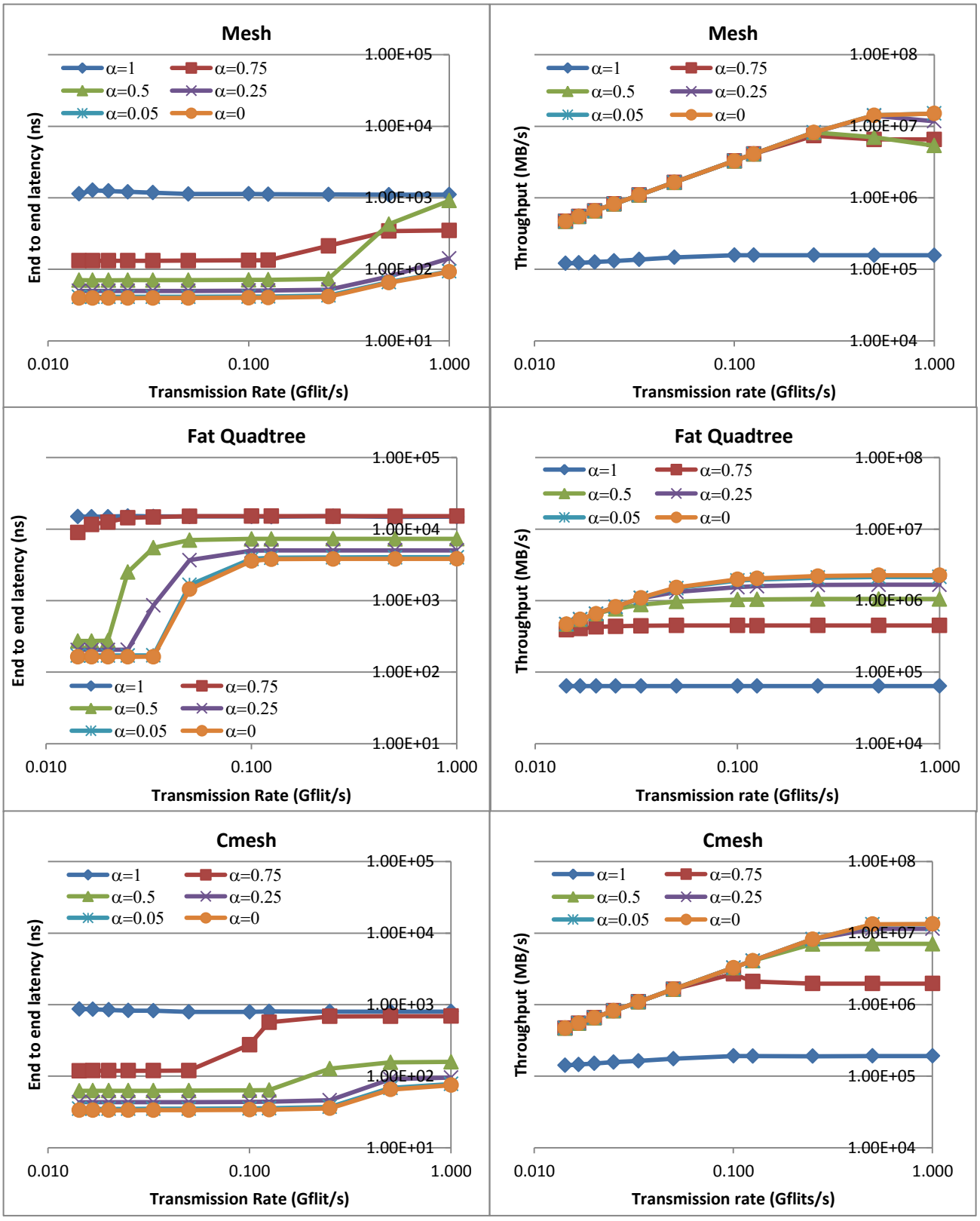

Figure 5. Group Clustering (α=1: no locality, α=0: total locality)

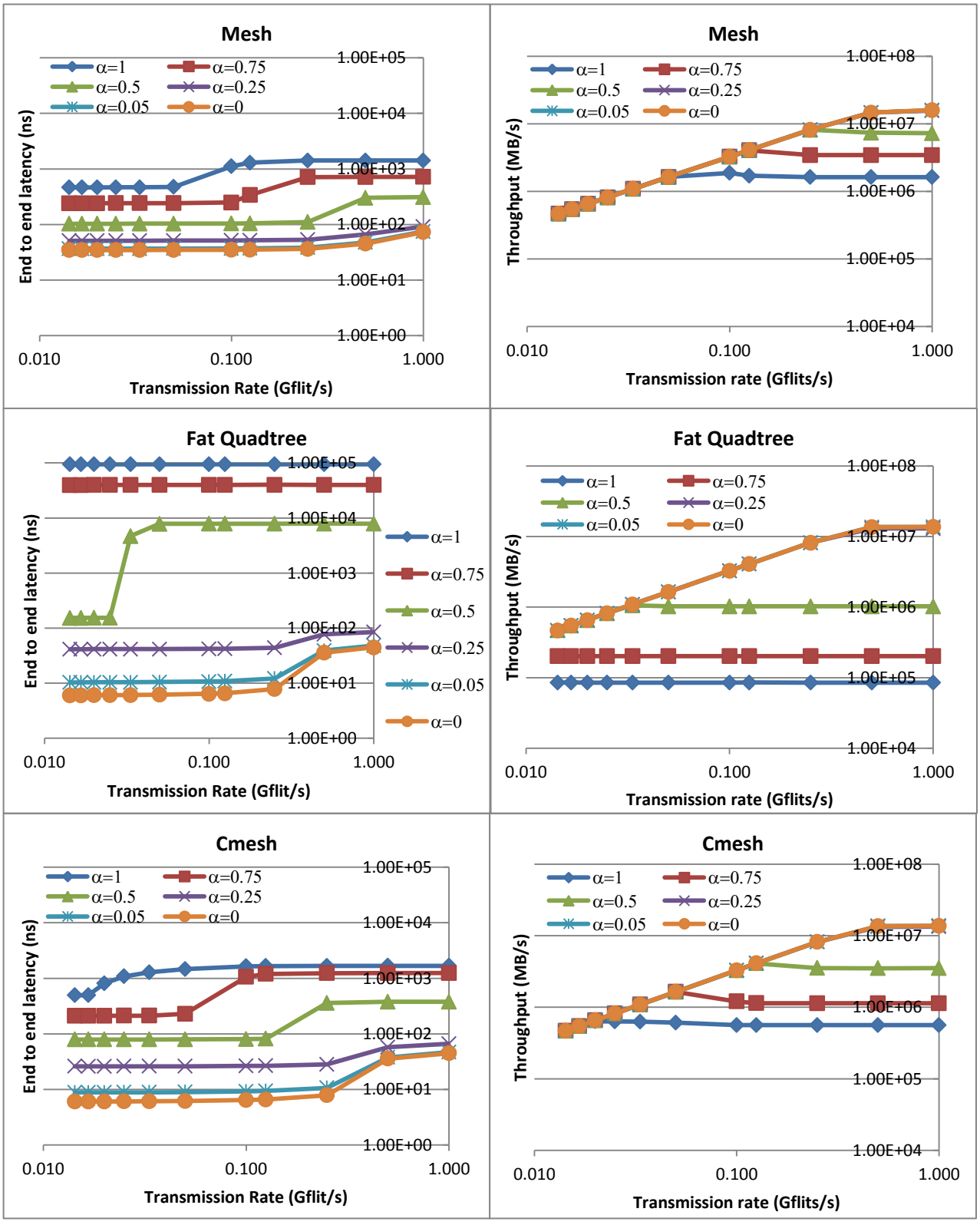

Figure 6. Ring Clustering (α=1: no locality, α=0: total locality)

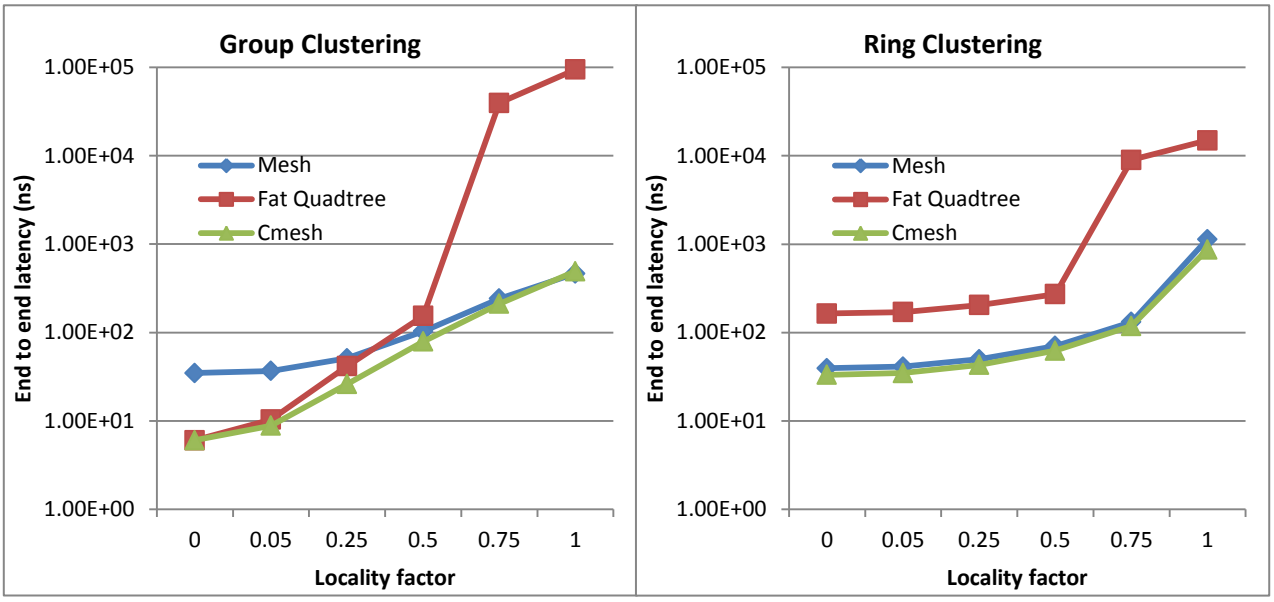

Figure 7. locality-based models (α=1: no locality, α=0: total locality)

Hence, we assume 256-bit wide flits, with a 10-bit address field for both the source and the destination. Using store-and-forward does not require the 2-bit markers for the flit type therefore the overhead is $2 \times 10/256 = 7.8\%$. By comparison, for the wormhole switching with 4 flits per packet as used in [20], the overhead is $(4 \times 2 + 2 \times 10)/(4 \times 256) = 2.7\%$.

The average packet latency of the system improves when reducing the number of flits per packet. The fat quadtree with single-flit packets performs 15% to 75% better compared to 4 flits per packet under high locality traffic ($\alpha < 0.25$). The reason is when more flits are in a packet more routers will be occupied by one packet which leads to increased latency. This effect is very clear in the fat quadtree topology and the gain in performance compensates for the 5% increased overhead for store-and-forward.

5. Evaluation and Discussion

We simulated the different topologies on 1024-core chip (placed in a regular 32×32 grid) using HNOCS (Heterogeneous Network-on-Chip Simulator) package, which is an open source NoC simulator [21] based on OMNeT++ [22]. OMNeT++ is an extensible, modular, open source component based C++ simulation library and framework, primarily aimed at building network simulators. The original HNOCS uses a mesh topology with wormhole switching with virtual channels and credit based flow control. It uses XY routing algorithm. We extended HNOCS with the concentrated mesh and fat quadtree topologies and their routing algorithms, as well as our locality-based traffic distributions.

Furthermore, we modified the switching system to accept single flit packets, in other words we added store-and-forward functionality.

In the simulation, all cores generate traffic at the same time. A core will generate the next flit only if the flit in the source queue was sent; hence there will be no dropped flits.

TABLE IV. Simulation Parameters

Topology	Mesh	CMesh	Fat quadtree
Virtual channels	2	2	0
Wire delay (ns)	17.5	35	$35 \times 2^{l-1}$, $1 \le l < n$
Flit size (bytes)	32		
Buffer size (flits/VC)	16		
Channel datarate (Gb/s)	128		

We model the process-to-process communication using Poisson-distributed traffic because it typically offers a good estimate on the average performance of networks and it has been widely used in the evaluation of interconnection networks. The packet length is one flit and the flit size is 32 Bytes. The channel data rate is 128 Gbps. Two virtual channels are used for the mesh and the concentrated mesh while for the fat quadtree one physical channel is used for the lowest-level links and it quadruples at each level to simulate a fat quadtree. Hence, the fat quadtree has no virtual channels. The buffer size in the router is 16 flits per virtual channel for the mesh and 16

flits per physical channel in the fat quadtree. The wire delay is proportional to the distance between the routers so in the fat quadtree it doubles at each level. The destination was selected using different degrees of localisation for each model. Table 4 summarises the simulation parameters used in our simulations.

We evaluated the performance of the three topologies: mesh, concentrated mesh and fat quadtree as a function of the transmission rate, the localisation degree α in two different locality-based models: group clustering and ring clustering. When $\alpha = 1$, it means there is no locality traffic, and when $\alpha = 0$ the traffic is fully local. Figures 5,6 and 7 show the results of our experiments in terms of latency and throughput.

For group clustering, the results show that the mesh has lower latencies when the degree of localisation is low $(0.5 \leq \alpha \leq 1)$ while fat quadtree has lower latencies when the degree of localisation is high $(0 \leq \alpha \leq 0.25)$. This indicates that the mesh will perform better when the traffic is uniformed and more distributed, however, the fat quadtree will perform better when the traffic is localised. It is known that the fat quadtree does not scale well, however, when locality is introduced it scales as figure 7 clearly show how the fat quadtree latencies improve when the degree of locality increases.

The concentrated mesh has low latencies when the traffic is highly localised traffic $(0 \leq \alpha \leq 0.25)$ similar to the latencies in the fat quadtree. This is because the concentrated mesh has four nodes for each router similar to the first level of the fat quadtree. The concentrated mesh has low latencies in low localised traffic $(0.5 \leq \alpha \leq 1)$ similar to the latencies in the mesh. This is because the concentrated mesh has similar structure to the mesh. However, the concentrated mesh congests faster than mesh because it has less links.

In terms of throughput, we observe that in nearly all the cases the throughput increases rapidly as the transmission rate increases. In case of low locality $(0.5 \leq \alpha \leq 1)$ the throughput is lower because in these cases the latencies were high and they are congested. For high locality $(0 \leq \alpha \leq 0.25)$, the throughput is nearly identical in all topologies.

Overall the concentrated mesh performs best in group clustering. One might expect the fat quadtree to perform better as the group clustering matches its topology. However, the layout of the fat quadtree results in fewer hops but longer paths, and in the 10 nm CMOS process for the 2023 node the wire delay is dominant (30 × worse than for the 2013 node).

For ring clustering, the mesh and the concentrated mesh have nearly similar latencies and they perform better than the fat quadtree. The fat quadtree has very high latencies as it congest fast. This is because in most cases the fat quadtree have fewer hop counts but longer paths with higher delays and it does not perform well when communicating with neighbouring nodes that do not have the same parent. In terms of throughput, the fat quadtree has lower throughput compared to the mesh and the concentrated mesh which have nearly identical throughput.

Overall, group clustering results in lower latencies than ring clustering; this is an important result for the placement of neighbours in stencil computations.

6. CONCLUSIONS

We have investigated the overhead and performance of flat (the mesh, the concentrated mesh) and scale-invariant (the fat quadtree) NoC topologies for future manycore systems with thousands of cores under group clustering and ring clustering localisation models. We show that the overhead of the NoC on a thousand-core system in 10 nm CMOS is negligible for all three topologies. We show that the degree of locality and the clustering model strongly affects the performance of the network. Scale-invariant topologies such as the fat quadtree perform worse than flat ones (esp. the concentrated mesh) because the reduced hop count is outweighed by the longer path delays, as a consequence of the high wire delay in the 10 nm CMOS process. Our results clearly show the importance of traffic localisation for very large manycore systems.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support provided by Sultanate of Oman, Ministry of Higher Education and the German University of Technology in Oman (GUTech).

REFERENCES

[1] Bell, Shane, Bruce Edwards, John Amann, Rich Conlin, Kevin Joyce, Vince Leung, John MacKay et al. "Tile64-processor: A 64-core soc with mesh interconnect." In *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pp. 88-598. IEEE, 2008.

[2] Duran, Alejandro, and Michael Klemm. "The Intel® many integrated core architecture." In *High Performance Computing and Simulation (HPCS), 2012 International Conference on*, pp. 365-366. IEEE, 2012.

[3] Loh, Gabriel H. "3D-stacked memory architectures for multi-core processors." In *ACM SIGARCH computer architecture news*, vol. 36, no. 3, pp. 453-464. IEEE Computer Society, 2008.

[4] Lim, Sung Kyu. "3D-MAPS: 3D massively parallel processor with stacked memory." In *Design for High Performance, Low Power, and Reliable 3D Integrated Circuits*, pp. 537-560. Springer New York, 2013.

[5] Armstrong, Joe. "erlang." *Communications of the ACM* 53, no. 9 (2010): 68-75.

[6] Moadeli, Mahmoud, Ali Shahrabi, Wim Vanderbauwhede, and Mohamed Ould-Khaoua. "An analytical performance model for the Spidergon NoC." In *Advanced Information Networking and Applications, 2007. AINA'07. 21st International Conference on*, pp. 1014-1021. IEEE, 2007.

[7] Moadeli, Mahmoud, Partha Maji, and Wim Vanderbauwhede. "Quarc: A High-Efficiency network on-Chip architecture." In *Advanced Information Networking and Applications, 2009. AINA'09. International Conference on*, pp. 98-105. IEEE, 2009.

[8] Balfour, James, and William J. Dally. "Design tradeoffs for tiled CMP on-chip networks." In *Proceedings of the 20th annual international conference on Supercomputing*, pp. 187-198. ACM, 2006.

[9] Leiserson, Charles E. "Fat-trees: universal networks for hardware-efficient supercomputing." *Computers, IEEE Transactions on* 100, no. 10 (1985): 892-901.

[10] Das, Reetuparna, Soumya Eachempati, Asit K. Mishra, Vijaykrishnan Narayanan, and Chita R. Das. "Design and evaluation of a hierarchical on-chip interconnect for next-generation CMPs." In *High Performance Computer Architecture, 2009. HPCA 2009. IEEE 15th International Symposium on*, pp. 175-186. IEEE, 2009.

[11] Pande, Partha Pratim, Cristian Grecu, Michael Jones, André Ivanov, and Res Saleh. "Effect of traffic localization on energy dissipation in NoC-based interconnect." In *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pp. 1774-1777. IEEE, 2005.

[12] Vanderbauwhede, Wim, and Tetsuya Takemi. "An investigation into the feasibility and benefits of GPU/multicore acceleration of the weather research and forecasting model." In *High Performance Computing and Simulation (HPCS), 2013 International Conference on*, pp. 482-489. IEEE, 2013.

[13] Greenfield, Daniel, Arnab Banerjee, Jeong-Gun Lee, and Simon Moore. "Implications of Rent's rule for NoC design and its fault-tolerance." In *Networks-on-Chip, 2007. NOCS 2007. First International Symposium on*, pp. 283-294. IEEE, 2007.

[14] Landman, Bernard S., and Roy L. Russo. "On a pin versus block relationship for partitions of logic graphs." *Computers, IEEE Transactions on* 100, no. 12 (1971): 1469-1479.

[15] Heirman, Wim, Joni Dambre, Dirk Stroobandt, and Jan Van Campenhout. "Rent's rule and parallel programs: characterizing network traffic behavior." In *Proceedings of the 2008 international workshop on System level interconnect prediction*, pp. 87-94. ACM, 2008.

[16] Howard, Jason, Saurabh Dighe, Sriram R. Vangal, Gregory Ruhl, Nitin Borkar, Shailendra Jain, Vasantha Erraguntla et al. "A 48-core IA-32 processor in 45 nm CMOS using on-die message-passing and DVFS for performance and power scaling." *Solid-State Circuits, IEEE Journal of* 46, no. 1 (2011): 173-183.

[17] Vangal, Sriram, Jason Howard, Gregory Ruhl, Saurabh Dighe, Howard Wilson, James Tschanz, David Finan et al. "An 80-tile 1.28 TFLOPS network-on-chip in 65nm CMOS." In *IEEE International Solid-State Circuits Conference, ISSCC 2007, Digest of Technical Papers, San Francisco, CA, USA*, pp. 98-99. IEEE, 2007.

[18] International technology roadmap for semiconductors (itrs) (2011).

[19] Killebrew, Carrell. "L2 Cache to Off-chip Memory Networks for Chip Multiprocessor." (2008).

[20] Lee, Junghee, Chrysostomos Nicopoulos, Sung Joo Park, Madhavan Swaminathan, and Jongman Kim. "Do we need wide flits in Networks-on-Chip?." In *VLSI (ISVLSI), 2013 IEEE Computer Society Annual Symposium on*, pp. 2-7. IEEE, 2013.

[21] Ben-Itzhak, Yaniv, Eitan Zahavi, Israel Cidon, and Avinoam Kolodny. "HNOCS: modular open-source simulator for heterogeneous NoCs." In *Embedded Computer Systems (SAMOS), 2012 International Conference on*, pp. 51-57. IEEE, 2012.

[22] Varga, András. "The OMNeT++ discrete event simulation system." In *Proceedings of the European simulation multiconference (ESM'2001)*, vol. 9, no. S 185, p. 65. sn, 2001.

Sharifa Al Khanjari is a PhD student in Embedded, Networked and Distributed Systems (ENDS) at the School of Computing Science of the University of Glasgow. She works as a lecturer in the German University of Technology in Oman (GUTech). She received a Master and Bachelor Degree in Computer Science from Sultan Qaboos University, Oman in 2007 and 2010, respectively. Her research focus on the architecture of many-core systems on network interconnect.

Dr Wim Vanderbauwhede is Lecturer in Embedded, Networked and Distributed Systems at the School of Computing Science of the University of Glasgow. His research focuses on high-level programming, compilation and architectures for heterogeneous manycore systems and FPGAs, with a special interest in power-efficient computing. He is author of the book "High-Performance Computing Using FPGAs". He received a PhD in Electrotechnical Engineering with Specialisation in Physics from the University of Gent, Belgium in 1996.