Complexity of Collective Communications on NoCs

Jiří Jaroš, Miloš Ohlídal, Václav Dvořák
Brno University of Technology
{Jarosjir, Ohlidal, Dvorak}@fit.vutbr.cz

Abstract

The paper addresses the important issue related to communication performance of Networks on Chip (NoCs), namely the complexity of collective communications measured by a required number of algorithmic steps. Three NoC topologies are investigated, a ring network, Octagon and 2D-mesh, due to their easy manufacturability on a chip. The lower complexity bounds are compared to real values obtained by evolution-based optimizing tools. Results give hints on what communication overhead is to be expected in ring- and mesh-based NoCs with the wormhole switching, full duplex links and k-port non-combining nodes.

1. Introduction

With an increasing number of processor cores, memory modules and other hardware units in SoCs, the importance of communication among them and of related interconnection networks is steadily growing. Recently the research opened up in Network on Chip (NoC) area, encompassing the interconnection /communication problem at all levels, from physical to the architectural to the OS and application level [1], [2].

Some embedded parallel applications, like network or media processors, are characterized by independent data streams or by a small amount of inter-process communications [1]. However, many general-purpose parallel applications display a bulk-synchronous behavior: the processing nodes access the network according to a global, structured communication pattern. They can, for example, execute a personalized all-to-all information exchange, global synchronization, gather/scatter to/from one node, etc. The performance of these collective communications (CC for short) has a dramatic impact on the overall efficiency of parallel processing. Provided that computation times are known, as is usually true in case of application-specific systems, the only thing that matters in obtaining the highest performance are durations of various collective communications.

Bus-based, synchronous communication structures in SoC, operating at several hundreds MHz, are not attractive any more due to tight timing constraints and skew control [2]. Transition to point-to-point high speed networks, that happened on system boards (e.g. from PCI to PCI/Express), is taking place on SoCs, too. Much research and practical interest has recently focused on other regular networks implemented on chip. A class of interconnection networks of interest in this paper covers direct networks, which for performance-driven environments converge on the use of pipelined (wormhole, WH) message transmission and source-based routing algorithms.

Logarithmic diameter networks, e.g. hypercube, butterfly and fat tree, provide enough bandwidth for all-to-all communications, but do not map well into two dimensions provided by a silicon chip: the length of some interconnection wires increases proportionally to the number of processors. This will decrease the clock frequency dramatically and degrade the performance. We therefore investigate here three NoC topologies with only local interconnection among processors, namely the ring, Octagon [3] and 2D-mesh.

The paper is structured as follows. In the following Section 2 we give the lower bounds on the number of communication steps for general networks and for our three topologies of choice. Then Section 3 deals with optimum CC algorithms that match the lower bounds or are close to them on 1-port and 2-port rings and on Octagon topology. Finally the all-port 2D-mesh is analyzed in Section 4. Results and their scalability are commented in Conclusions. We will use P for number of processors in the network with V nodes (vertices). If $P=V$, we have "slim" nodes - one processor per node; otherwise we can place P/V processors on one node and get "fat" nodes.

2. Time complexity of collective communications in WH networks

Performance of CCs is closely related to their time complexity. The simplest time model of point-to-point communication in direct WH networks takes the communication time composed of a fixed start-up time t_S at the beginning (SW and HW overhead) and of a component that is a function of distance h (the number of channels on the route or hops a message has to do) and message length m in certain units (words or bytes):

$$t_{WH} = t_S + h\,t_r + m\,t_1, \quad (1)$$

where t_r includes a routing decision delay, switching and inter-router latency and t_1 is per unit-message transfer time. The dependence on h is rather small, (since $t_r << m\,t_1$), so that WH switching is considered distance-insensitive. For simplicity, in eq. (1) we have assumed no contention for channels and no associated delays.

Beside pair-wise communications, in many parallel algorithms we often find certain communication patterns, which are regular in time, in space, or in both time and space; by space we understand spatial distribution of processes on processors. Communications taking place among a subset or among all processors are called collective (CC) or group communications. Generally we have two sets of nodes: T – the set of transmitting nodes and R – the set of receiving nodes. We may distinguish three classes of CCs:

1. $T \cap R = \varnothing$, non-overlapping sets of nodes.

   A. One-to-all, $|T| = 1$, $|R| = P$-1. Broadcast communication (OAB, a single message) belongs to this class as well as one-to-all scatter communication (OAS, a private message to each partner).

   B. All-to-one, $|T| = P$-1, $|R| = 1$, e.g. gather (AOG) or reduce (AOR) communication.

   C. Many-to-many, $|T| = M$, $|R| = N$. Non-overlapping sets of nodes.

2. $|T \cap R| \geq 2$. Many-to-many communication with overlapping sets of nodes.

3. $|T \cap R| = P$. All-to-all communications such as permutation, all-to-all scatter, (AAS), all-to-all reduce (AAR), and others.

Since complexities of some communications are similar (AOG ~ OAS, AOR ~ OAB, AAR ~ AAB), we will focus only on 4 basic types (OAB, OAS, AAB, AAS). Also, from now on, when we refer to „collective communications", then we will assume only CCs involving all processors.

In the rest of the paper we assume that the CC in WH networks proceeds in synchronized steps. In one step of CC, a set of simultaneous packet transfers takes place along complete disjoint paths between source-destination node pairs. If the source and destination nodes are not adjacent, the messages go via some intermediate nodes, but processors in these nodes are not aware of it; the messages are routed automatically by the routers attached to processors. Complexity of collective communication will be determined in terms of the number of communication steps or equivalently by the number of "start-ups" $\tau_{CC}(G)$ (lower bound). This figure of merit does not take into account the message length or its variations from one step to another.

The number of CPU ports on every node that can be engaged in one step of CC will be denoted k, meaning that $2k$ (DMA) unidirectional channels (k inputs, k outputs to/from the CPU) between the processor and an associate router can transfer data simultaneously. Always $k \leq d$, where d is a node degree; 1-port model (k=1) and the all-port router model (k=d) are most frequently used. Parameter k has also an impact on number of communication steps, as well as the fact if nodes can combine/extract partial messages with negligible overhead (combining model) or can only re-transmit/consume original messages (non-combining model). Finally, the number of steps $\tau_{CC}(G)$ depends on a channel type; we have to distinguish between unidirectional (simplex) channels and bi-directional (half-duplex HD, full-duplex FD) channels. Typically $\tau_{CC}(G)$ will be 2-times larger for HD channels than for the FD ones. Further on we will consider only FD channels.

One of the key design factors of an interconnection network is its topology. The lower bounds $\tau_{CC}(G)$ for the network graph G depend on node degree d, number of nodes P, and bisection width B_C, Tab.1.

As far as the broadcast communication (OAB) is concerned, the lower bound on the number of steps $\tau_{OAB}(G) = s = \lceil \log_{k+1} P \rceil$ is given by the number of nodes informed in each step, that is initially 1, $1+1\times k$ after the first step, $(k+1)+(k+1)\times k = (k+1)^2$ after the second step, etc.,…, and $(k+1)^s \geq P$ nodes after step s.

In case of AAB communication, since each node has to accept $P-1$ distinct messages, the lower bound is $\lceil (P-1)/k \rceil$ steps. A similar bound applies to OAS communication, because each node can inject into the network not more than k messages in one step; for irre-

Table 1. Lower bounds on complexity of collective communications

CC	WH, k-port, FD model
OAB	$\lceil \log_{k+1} P \rceil = \lceil (\log P)/\log(k+1) \rceil$
AAB	$\lceil (P-1)/k \rceil$
OAS	$\lceil (P-1)/k \rceil$
AAS	$\lceil d/k \rceil \lceil P^2/(2\,B_C) \rceil$

gular networks with non-constant node degree d we should use the lowest value of k for AAB and the value of k of the source node for OAS. In WH OAS we use a broadcast tree for $P-1$ pair-wise communications, k of them per step. Apparently, to pack k-tuples of messages into the lowest number of steps, an optimum broadcast tree should have k sub-trees of approximately the same size. All k paths traversed in each step must be link-disjoint to avoid conflicts.

The lower bound for AAS can be obtained considering that one half of messages from each processor cross the bisection, whereas the other half do not. There will be altogether $\lceil 2(P/2)(P/2)/\,B_C \rceil$ of such messages in both ways, where B_C is the network bisection width [4]. This gives $\lceil d/k \rceil \lceil P^2/(2B_C) \rceil$ steps if k communications can run in parallel.

For the network topologies potentially useful in NoC the lower bounds of selected CCs are given in Tab.2. For regular networks (with constant node degree d) k-port model is specified. 2D meshes have 3 values of k, k=2, 3 and 4 and one-to-all communications depend on the k-value of the source node. Wormhole switching and full duplex links are assumed.

Table 2. Lower bounds $\tau_{CC}(G)$ from Tab.1 for selected networks

WH, FD, k-port	OAB	AAB	OAS	AAS
Ring 8, k=1	3	7	7	16
Ring16, k=2	4	15	15	64
Ring 8, k=2	2	4	4	8
Ring 16, k=2	3	8	8	32
Octagon 8, k=3	2	3	3	3
16-gon, k=3	2	5	5	11
2D mesh 4 x 2	2	4	2, 4	8
2D mesh 4 x 4	2,2,3	8	4,5,8	16
2D mesh 6 x 6	3,3,4	18	9,12,18	54
2D mesh 8 x 8	3,3,4	32	16,21,32	128

3. Complexity of real CC algorithms on ring topologies

The bidirectional ring topology, though very simple, is not free from routing deadlock, because the channel dependency graph is not acyclic [4]. This can be seen on a 1-port as well all-port (2-port) ring on a common permutation called cyclic shift. The problem can be solved by introduction of virtual channels [4] and by implementing rules on channel usage. We assume that these rules are adhered to in all our CC schedules and thus the deadlock is avoided.

As far as 2D meshes is concerned, the dimension-ordered deterministic routing (first in x, then in y direction) on meshes and tori is known to be deadlock-free. A certain degree of adaptivness can be obtained by more relaxed routing, such as North-last or West-first strategy [4].

3.1. CCs on an 1-port and all-port ring

The optimal OAB algorithm reaching the lower bound $\tau_{OAB}(G) = \lceil \log_{k+1} P \rceil$ recursively doubles (k=1) or triples (k=2) the number of informed nodes in each step. The ring is split into 2 halves (k=1) or 3 thirds (k=2) and then the source sends a message to its image in the other half or images in other thirds.

OAS algorithm spreading customized messages to every partner is trivial, because the source has to inject 1 (k=1) or 2 (k=2) messages at a time into the ring and the lower bound clearly applies.

AAB communication among 1-port nodes also has a straightforward solution: all processors just send their messages in one direction around the ring and the communication proceeds by pipelining in P-1 steps. All the channels are used in all steps. In 2-port model, every node sends its messages into two branches of a primitive broadcast tree and they do one hop in each step. It is easily seen that these broadcast trees of all the nodes are time-arc disjoint, i.e. no channel is used more than once in a single step. Thus the lower bound $\tau_{OAB}(G)=\lceil (P-1)/2 \rceil$ can again be reached.

The last AAS communication could be implemented as $(P-1)$ permutations, deadlock-free with virtual channels, but not without link congestions (conflicts). Communication time thus cannot be estimated. We have therefore tried to find AAS communication schedule organized into congestion-free steps with the use of evolutionary optimization. We have used the Mixed Bayesian Optimization Algorithm (MBOA) [5], which is based on BOA (Bayesian Optimization Algorithm). The probabilistic model of BOA, the

Bayesian net, is replaced by a set of binary decision trees/graphs.

A chosen AAS chromosome encoding has a form of a matrix with P OAS chromosomes (vectors). The OAS chromosome uses P genes, each gene consists of two items: an index of one of the shortest source-destination path and a communication step number. The fitness function is based on counting conflicts in schedules (i.e. situations when two processors want to use the same channel in the same step). The optimal schedule does not contain any conflict and the MBOA (with the given number of communication steps as input) was able to find them for common networks with up to 64 nodes [5].

Table 3. AAS communication schedule on the 8-node bidirectional all-port ring

step	clockwise from >to
	counter-clockwise
0	1 >5, 5 >1
	1 >6, 6 >5, 5 >3, 3 >1
1	0 >1, 1 >4, 4 >5, 5 >0
	0 >6, 6 >4, 4 >2, 2 >0
2	0 >3, 3 >7, 7 >0
	0 >5, 5 >4, 4 >1, 1 >0
3	1 >3, 3 >6, 6 >1
	0 >4, 4 >3, 3 >0
4	0 >2, 2 >4, 4 >6, 6 >0
	0 >7, 7 >4, 4 >0
5	1>2, 2>3,3 >5, 5>7, 7>1
	2 >6, 6 >3, 3 >2
6	2 >5, 5 >6, 6 >7, 7 >2
	1 >7, 7 >6, 6 >2, 2 >1
7	3 >4, 4 >7, 7 >3
	2 >7, 7 >5, 5 >2

A sample solution of optimal AAS schedule for the bidirectional ring of 8 processors is shown in Tab.3. The number of steps reaches the lower bound τ_{AAS}(Ring) = 8 and 56 messages get distributed. In each step one or two processors use all (two) their ports, i.e. two pairs of channels connecting them to the router. E.g. in step 1, processors 0 and 4 communicate simultaneously on 2 ports (2 input and 2 output channels each). Let us note that several optimal solutions exist. For HD links or 1-port nodes the number of steps would be double, routing clockwise and counter-clockwise direction in separate steps.

3.2. Scheduling CCs on all-port Octagon network

Octagon is the novel on-chip communication network architecture suitable for the aggressive on-chip communication demands of SoCs in several application domains and also for networking SoCs [3], Fig.1. As a ring, it is also not free from deadlock and virtual channels have to be used. The suggested scaling strategy [3] based on bridge nodes connecting adjacent Octagons has a drawback of a very low bisection width B_C and therefore a poor performance in all-to-all traffic. Another scaling strategy extends the Octagon to the multidimensional space by linking corresponding nodes of several Octagons. This, however, increases the node degree, and is not always acceptable. We will therefore use a generic NoC model with P = 8, 12, 16, …, 4n retaining the original topology [7].

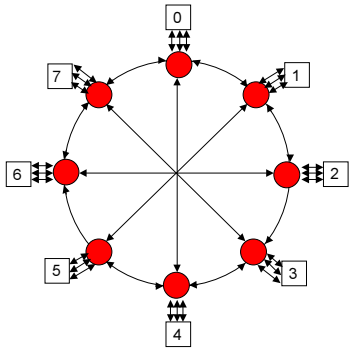

Fig.1. Octagon network, all-port model (k = 3)

Collective communications on the generic 8-processor, symmetric Octagon network are easy. One-to-all communications are done the same way for every source node. OAB clearly can be done in 2 steps and OAS needs $\lceil 7/3 \rceil$ = 3 steps. To implement AAB, we have to use such a broadcasting tree that is time-arc-disjoint (TADT) and can be used by all nodes simultaneously without creating conflict. The same tree as for store and forward switching can be used, restricting communication in each step to only between neighbors. E.g. node 0 could use this TADT:

Step 1: 0 >7, 0 >4, 0 >1
Step 2: 7 >6, 1 >2
Step 3: 4 >5, 4 > 3.

We cannot join steps 2 and 3 though, because it would create conflict - one node cannot use more than 3

channels in a single step, because there is not more than 24 channels altogether.

The most complex AAS communication is not performed the same way by all nodes - there is no analogy to the TADT. In the design of AAS schedule, the same approach has been used as with the bidirectional ring, namely evolutionary optimization. Four steps were needed for AAS on Octagon with all-port (3-port) nodes, one step worse than the lower bound in Tab.1. The optimum AAS schedule is given in Tab. 4. The sequences of digits denote the path of length one (src, dst) or two (src, via, dst). For a scalable version of Octagon with the number of nodes increased by 4, we can find upper bounds similarly as for Octagon using evolutionary algorithm. The results for 16-gon ("hexadecagon") are given in Table 5. The cases where the upper bound differs from the lower bound are denoted by bold digits.

Table 4. AAS communication schedule on the Octagon8 topology

step	AAS on Octagon
0	073, 104, 156, 21, 23, 267, 340, 432, 45, 512, 654, 701, 762
1	012, 07, 10, 265, 321, 34, 451, 437, 54, 567, 623, 73, 704, 76
2	01, 12, 15, 107, 234, 26, 32, 40, 456, 543, 670, 621, 765
3	04, 015, 076, 123, 210, 345, 326, 37, 43, 540, 51, 56, 62, 65, 67, 70

4. Real CC algorithms on 2D meshes

2D-meshes may be easy to manufacture on a chip due to local interconnections only, but they have other disadvantages. The main one is a lack of node symmetry. Meshes are also irregular networks as the node degree is not constant. While the corner nodes have degree 2, the nodes on the boundary have 3 links and internal nodes 4 links. Therefore one-to-many algorithms for all-port meshes ($k = d$) will need more or less steps accordingly. This is a big difference in comparison to node symmetric tori networks.

OAB in 1-port 2D-meshes is relatively easy task: recursive doubling (as on the ring) is done first in x dimension and as soon as all nodes in row x are informed, we do OAB in all columns simultaneously, again by recursive doubling.

On the contrary, to develop optimal OAB algorithm in all-port 2D-meshes is difficult, because the lower bound is pretty tight. To achieve it, we need an algorithm, in which every node, once informed, must find in every subsequent step 4 uninformed nodes and deliver them the message, so that globally all used paths are link-disjoint. Since meshes, unlike tori, are not node-symmetric, there are no elegant algorithms for them. Several approaches have been developed for tori; e.g. torus is split into 5 horizontal strips, the message is sent from the source strip to all other strips in step 1 using x then y routing, and then recursively the same in each horizontal strip. This would not work for small tori and even worse so for small meshes. Here again we have to resort to evolutionary or other kind of optimization. The results are shown in Tab.5.

The similar situation is in design of TADT trees useful for OAS and AAB communications. They are known for square tori, but in the other cases their construction is difficult, due to the lack of symmetry. Finally, the congestion- (conflict) free AAS schedules are not known even for square tori.

The design of CC schedules using evolutionary algorithms has been carried out in two directions:
- OAB, OAS and AAS schedules were obtained as already presented above with the aid of MBOA.
- AAB schedules were tackled with Hybrid parallel Genetic Simulated Annealing (HGSA) [6].

In HGSA, there are sequential SA processes running in parallel. After 100 or so iterations of Metropolis algorithm, each process sends its solution to a master. The master uses the genetic crossover to produce new solutions: two children solutions are generated from two parents by means of a genetic crossover. Then the mutation is performed (always in case of the parent solution, otherwise only with a predefined probability). Based on the roulette wheel, master selects randomly one solution from the new generation for itself and other solutions that it sends to slaves (one per slave).

The results of MBOA and HGSA optimization are summarized in Tab.5. Bold digits represent cases when lower bounds were not reached. With exception of AAS on 16-gon, the difference is in remaining cases just one step. Of course, the fact that the lower bound cannot be always reached is to be expected and no other algorithm can ever change it. In some cases it was verified that the difference of 1 step is due to the minimum routing strategy used in evolutionary algorithms. Inclusion of the non-minimum routing would lead to an enormous increase of possible paths from sources to destinations and therefore was not explored. However, in some small networks the analysis of the last remaining conflict in fitness function revealed, that it can be eliminated if non-minimum routing is used. This way the upper and lower bounds were made equal (optimum schedules).

5. Conclusions

The lower and upper bounds on number of CC steps, denoted $\tau_{CC}(G)$ and $\tau^{CC}(G)$, were presented for interconnection networks (G) of interest for NoCs. Since a distance-insensitive wormhole switching was assumed, the real communication times can be obtained approximately as number of start-ups plus the serialization delay $m\,t_1$,

$$t_{CC} = t_S \times \tau^{CC}(G) + m\,t_1, \qquad (2)$$

if neglecting the hardware overhead in routers along the traversed path. Possible synchronization overhead involved in communication steps, be it hardware or software-based, should be included in the start-up time t_S. According to frequency of CCs and an amount of interleaved computation in a certain application, efficiency of parallel processing can be estimated.

Table 5. Real complexities of CC on selected k-port networks – upper bounds $\tau^{CC}(G)$

WH, FD	OAB	AAB	OAS	AAS
Ring 8, k=1	3	7	7	16
Ring16, k=1	4	15	15	64
Ring 8, k=2	2	4	4	8
Ring 16, k=2	3	8	8	32
Octagon, k=3	2	3	3	**4**
16-gon, k=3	2	**9**	5	**17**
2D mesh 4 x 2	**4***)	4	4	8
2D mesh 4 x 4	2,2,3	8	4, **6**, 8	**17**
2D mesh 6 x 6	3,3,4	18	9,12,18	+)
2D mesh 8 x 8	3,**4**,4	**33**	16,**22**,32	+)

*) 3 steps with non-minimum routing
+) not completed as yet

It is seen from the results, that even though the upper bounds are mostly close or equal to lower bounds, scalability of CC algorithms, especially of all-to-all communication, on networks being considered as candidates for NoCs, is not too good. The reason is that these networks, simple enough to be manufactured easily, do not have enough bisection bandwidth or sufficient port model. If we look for example at 1-port rings and AAS pattern, we could implement it with not more than P-1 permutations. From Tab. 5 we can see that each such permutation would have to generate in average link congestion 2 and 4 (the same link used 2- and 4-times in a single permutation). In this respect the AAB communication is easier, because only one message per processor may have to cross the bisection.

One way of performance improvement is to scale small generic networks to fat networks (with multiple processors per node and/or multiple edges) and provide more ports for simultaneous communication. However, this approach is not cheap in hardware, although manufacturability remains easy.

Future research will be oriented towards optimization of CCs in fat networks, whose potential for NoCs was not yet fully appreciated. Another direction worth of effort is a class of many-to-many CC with non-overlapping as well as overlapping subsets of processors. The application-specific CCs of this kind are of increasing importance on multiprocessor SoCs.

6. References

[1] A. Jantsch, H. Tenhunen, *Networks on Chip*, Kluwer Academic Publ., Boston, 2003.

[2] A. Ivanov, G. De Micheli, "Guest Editors' Introduction: The Network-on-Chip Paradigm in Practice and Research", *IEEE Design&Test of Computers*, IEEE Los Alamitos CA, Sept.-Oct. 2005, pp. 399-403.

[3] Karim, F., Nguyen, A.: An Interconnect Architecture for Networking Systems on Chips. *IEEE Micro*, Sept. – Oct. 2002, pp.36-45.

[4] Duato, J., Yalamanchili, S.: *Interconnection Networks – An Engineering Approach*, Morgan Kaufman Publishers, Elsevier Science, 2003

[5] Ocenasek, J.: Parallel Estimation of Distribution Algorithms, PhD. Thesis, Faculty of Information Technology, Brno University of Technology, Brno, Czech Rep., 2002.

[6] Jaroš, J., Ohlidal, M., Dvorak, V.: Evolutionary Design of Group Communication Schedules for Interconnection Networks. *Lecture Notes in Computer Sciences* 3733, Berlin, DE, Springer, 2005, s. 472-481.

[7] Schmaltz, J., Borrione, D.: A Generic On Chip Network Model. *Tima Lab. Research Report* ISRN TIMA-RR-05/03-06-FR, 2005.

Acknowledgement

This research has been carried out under the financial support of the research grant "Network Architectures of Embedded Systems Networks", GA102/05/0467, Grant Agency of Czech Republic, 2005-2007.