Optimum Topology-Aware Scheduling of Many-to-Many Collective Communications

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

Abstract

The paper addresses general many-to-many collective communications, whose scheduling may be needed when writing application-specific communication routines or communication libraries. Optimum schedules with the number of steps equal or close to theoretical lower bounds are designed with the use of evolutionary algorithms. Optimization is carried out for a given topology of a direct interconnection network; network nodes can be single or multiple processors connected to a router. Wormhole switching, full duplex links and single-port non-combining nodes are assumed. The developed scheduling could be advantageous mainly for networks on chip (NoC) and application-specific communication architectures.

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 is duration 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; that is why only wormhole switching is considered in this paper.

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. In this paper we use Octagon topology [3] and 2D-mesh for illustration, which could be used as NoCs.

The paper is structured as follows. In the following Section 2 we review the lower bounds on the number of communication steps in WH networks and for All/One-to-All/One communication patterns, whereas in Section 3 we give new bounds for M-to-N communication patterns. In Section 4 the CC scheduling problem is formulated mathematically and methods of its solution are discussed. Our approach is explained in Section 5. Finally Section 6 is a small case study comparing upper and lower bounds for selected communications on 1-port fat Octagon topology. 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 All/One-to-All/One 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):

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

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 (and therefore congestion, too) 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 subsets or among all processors are called collective (CC) or group communications. We will assume that all messages in CC have identical size. 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. $T \cap R = \emptyset$, 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, $|T| = M$, $|\mathrm{R}| = N$. Non-overlapping sets of nodes.

2. $|T \cap R| \geq 1$. 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.

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 (frames) or equivalently by the number of "start-ups"; there are two figures - theoretical lower bound $\tau_{CC}(G)$ or really obtainable upper bound $\tau^{CC}(G)$. These figures of merit do not take into account the message length or its variations from one step to another. Further on we assume that the CPU is connected to an associate router via two unidirectional DMA channels (one-port model), which can transfer data simultaneously (full duplex). A more general k-port model would allow $2k$ such DMA 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 channel bisection width B_C, [8], Tab.1. Complexities of collective communications of type 1A, 1B and 3 in Table 1 are well known. As far as the broadcast communication (OAB) is concerned, the lower bound on the number of steps

$$\tau_{OAB}(G) = s = \lceil \log_2 P \rceil$$

is given by the number of nodes informed in each step, that is initially 1, 1+1 after the first step, 2 + 2 = 2^2 after the second step, etc.,…, and $2^s \geq P$ nodes after step s. In case of AAB communication, since each node has to accept P–1 distinct messages, the lower bound is P–1 steps. A similar bound applies to OAS communication, because each node can inject into the network not more than one message at a time.

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 $2(P/2)(P/2)$ of such messages in both ways. If B_C is the network bisection width [8], not more than B_C messages can flow in one direction through the cut at a time. This gives $\lceil P^2 /(2B_C) \rceil$ communication steps.

Table 1. Lower bounds on complexity of selected collective communications

CC	WH, 1-port, FD model
OAB	$\lceil \log_2 P \rceil$
AAB	$P-1$
OAS	$P-1$
AAS	$\max(\lceil P^2/(2B_C) \rceil, P-1)$

3. Time complexity of M-to-N collective communications in WH networks

The cases of M-to-N broadcast and scatter communication are represented in Fig.1. The sets of transmitting nodes T and receiving nodes R are generally overlapping, $|T \cap R| = Q \geq 1$, some nodes are only transmitting, $|T \setminus T \cap R| = M - Q$ and some nodes only receiving $|R \setminus (T \cap R)| = N - Q$. In special case $Q = R$ or $Q = T$, Fig.1b, c. The lower bounds $\tau_{CC}(G)$ for M to N communications are not known, but have been derived here as follows.

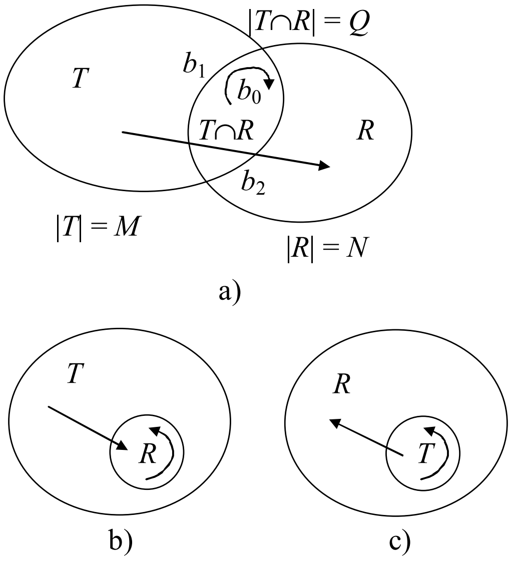

Fig.1. Many-to-many communication a) $T \not\subset R$ and $R \not\subset T$, b) $R \subset T$, c) $T \subset R$

Case a. First, M-to-N broadcast is limited by OAB or AAB bound from Tab.1, whichever is greater:

$$T_{MNB} = \max(\lceil \log_2 N \rceil, M) \qquad (2)$$

because some nodes may absorb M and not M–1 messages as in AAB.

Second, M-to-N scatter communication can be divided into four groups of communication that have related bisection widths available, see Fig.1:

$$\begin{array}{ll} T \setminus (T \cap R) \rightarrow (T \cap R) & b_1 \\ (T \cap R) \rightarrow R \setminus (T \cap R) & b_2, \\ T \setminus (T \cap R) \rightarrow R \setminus (T \cap R) & \min(b_1, b_2), \\ (T \cap R) \rightarrow (T \cap R) & b_0. \end{array} \qquad (3)$$

Now the first two groups may proceed simultaneously (overlapped),

$$T \setminus (T \cap R) \rightarrow (T \cap R) \parallel (T \cap R) \rightarrow R \setminus (T \cap R) \qquad (4)$$

and so could the other two:

$$T \setminus (T \cap R) \rightarrow R \setminus (T \cap R) \parallel (T \cap R) \rightarrow (T \cap R). \qquad (5)$$

Time for communication specified in (4) is thus

$$T_1 = \max\left(\left\lceil \frac{(M-Q)Q}{b_1} \right\rceil, \left\lceil \frac{Q(N-Q)}{b_2} \right\rceil \right)$$

and for communication described in (5) is

$$T_2 = \max\left(\left\lceil \frac{(M-Q)(N-Q)}{\min(b_1, b_2)} \right\rceil, \left\lceil \frac{Q(Q-1)}{2b_0} \right\rceil \right).$$

The total lower bound is thus

$$T_{MNS}(G) = T_1 + T_2. \qquad (6)$$

Case b. Similarly as in case a) for broadcast we have

$$T_{PNB} = \max(\lceil \log_2 P \rceil, P-1) \qquad (7)$$

and for scatter

$$T_{PNS} = \max\left(\left\lceil \frac{(P-N)N}{b_1} \right\rceil, \left\lceil \frac{N^2}{2b_0} \right\rceil \right), \qquad (8)$$

where b_1 is the number of channels from T to R and b_0 is a bisection width of sub-network R.

Case c. A similar reasoning as in case b gives

$$T_{MPB}(G) = \max(\lceil \log_2 P \rceil, M) \qquad (9)$$

and

$$T_{MPS} = \max\left(\left\lceil \frac{M(P-M)}{b_1} \right\rceil, \left\lceil \frac{M^2}{2b_0} \right\rceil \right), \qquad (10)$$

where b_1 is the number of channels from T to R and b_0 is a bisection width of sub-network T. In any case, the number of messages injected to or absorbed from the network in one step is limited also by the number of ports.

To illustrate the above results, M-to-N broadcast on the mesh in Fig.2 will be analyzed first. We have M = 9 sending nodes, N = 11 receiving nodes, and Q = 4 nodes in intersection $T \cap R$. According to eq. (2),

$$T_{MNB} = \max(\lceil \log_2 11 \rceil, 9) = \max(4, 9) = 9 \text{ steps}.$$

The complexity of scatter communication will be estimated by means of bisections $b_1 = 5$, $b_2 = 6$, and $b_0 = 1$. Substituting these parameters into eq. (6) we get

$$T_{MNS}(G) = T_1 + T_2 = \max(4, 5) + \max(7, 6) = 12 \text{ steps}.$$

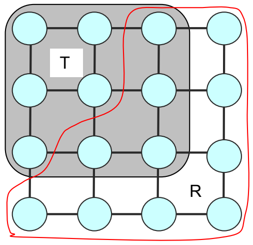

Fig.2. Nine-to-eleven scatter communication

This bound does not take into account a node degree and is therefore valid generally for *k*-port model, when CPU can communicate with the router via *k* ports simultaneously.

In the case when sets *T* a *R* are not overlapping, $T \cap R = \emptyset$, and $| T \cup R | \leq P$, we cannot use the bisection width any longer. If $\lambda_{CC}(l)$ is the load of link *l* in CC (i.e. the number of messages using link *l*) then the lower bound is given as

$$T_{M2N} = \max \lambda_{CC}(l) \qquad (11)$$

over all links *l*.

4. Mathematical formulation of the scheduling problem and its solutions

Any collective communication is composed of a set *CC* of pair-wise communications (transfers, messages, paths)

$$x_i = \{c_{src}, c_a, c_b, \ldots, c_{dst}\},$$

where c_x are channels along the minimum path from the source to destination node. (We will restrict ourselves to minimum routing for practical reasons given later). Cardinality of set *CC* may be quite high, e.g. all-to-all communication among *P* processors gives $|CC| = P(P-1)$ messages; for $P \in$ <8, 128> we have $P(P\text{-}1) \in$ < 56, 16256 >.

The goal of scheduling is to pack messages in *CC* into the minimum number of groups such, that there is no conflict within a group. In wormhole routing a conflict means that two messages scheduled in the same step share one or more channels. If they don't, they are compatible. Compatibility relation γ on set *CC* can thus be defined:

$$x_i \gamma x_k \equiv \exists! c_e \{c_e \in x_i \text{ and } c_e \in x_k\}.$$

This relation defines a cover of *CC* by maximum-size compatibility classes. A group of messages in one compatibility class can start transmission simultaneously and we therefore schedule each such group in one communication step. Obviously we want to find a minimum number of compatibility classes still covering set *CC*. The final step is to transform this minimum cover of *CC* to a partition, compatibility classes to blocks, by eliminating messages in more than one class and possibly simultaneously balancing the size of classes.

Exact solution of the above problem can be obtained by MILP method (Mixed Integer Linear Programming), but very long solutions are required for network sizes of practical interest. The communication scheduling can also be formulated as a graph coloring problem. Elements of *CC* can be represented by nodes in a graph, and conflicts among elements by graph edges. Minimum number of colors needed to color the graph gives the number of communication steps and nodes with the same color belong to one compatibility class. Exact or heuristic graph coloring, even though it may be quite lengthy, yields only a suboptimal solution. The reason is the existence of multiple minimum paths for some source-destination pairs. Which one should be selected? Another approach, recursive division of a *CC* set described in [4], is supposed to be exact, but has the following restrictions:

- only multicast is solved,
- routing from src to dst is unique and prescribed,
- one-port model is assumed.

In our approach (Section 5), we will relax all above restrictions.

Let us note that during the search for the optimum schedule, it may be necessary to include not only multiple minimum paths, but sometimes even non-minimum ones! Fig. 3 shows one example – one-to-all scatter communication in the mesh topology. To reach the minimum number of communication steps (the lower bound is 5 steps), 3 messages must be injected to a network in every step by the source node. The last step requires non-minimum routing.

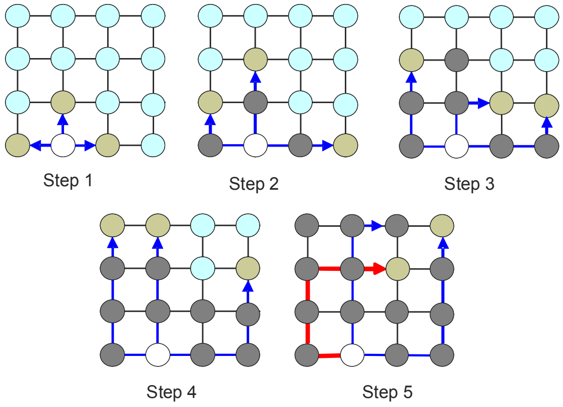

Fig. 3. One-to-all scatter in 5 steps

5. Evolutionary Search for Suboptimal Schedules

The design of conflict-free schedules using evolutionary optimization has been carried out in two directions:

- MNB and MNS schedules were obtained with the aid of MBOA algorithm.
- MNB schedules were tackled with Hybrid parallel Genetic Simulated Annealing (HGSA) [6].

Mixed Bayesian Optimization Algorithm (MBOA) [5] is based on 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 a schedule (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 it for common networks with up to 64 nodes [5].

In HGSA, there are sequential SA (Simulated Annealing) 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 MBOA and HGSA optimization were applied successfully to several network topologies. 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 one case it was verified that the non-minimum routing does make a difference (7 vs. 10 steps). However, generally only the minimum routing strategy was adopted in evolutionary algorithms because inclusion of the non-minimum routing would lead to an enormous increase of possible paths from sources to destinations and to the prohibitive memory and time consumption.

The situation was different with regard to multiple minimum paths from source to destination nodes. They were accounted for an easy way through mutation. As soon as the fitness was not improving in a certain period of optimization, replacement of one minimum path by another was a good remedy.

6. The Case Study: Real and Theoretical Complexity of M-to-N Communications on a Fat Octagon

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]. 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 and M-to-N 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. Octagon can also be extended to a larger ring with P = 8, 12, 16,…, 4*n* nodes retaining the original topology [7], but congestion of wires in the middle may cause difficulties at manufacturing (in 2 dimensions, e.g. in NoC). We have therefore used a fat Octagon with two CPU cores per node, Fig. 4.

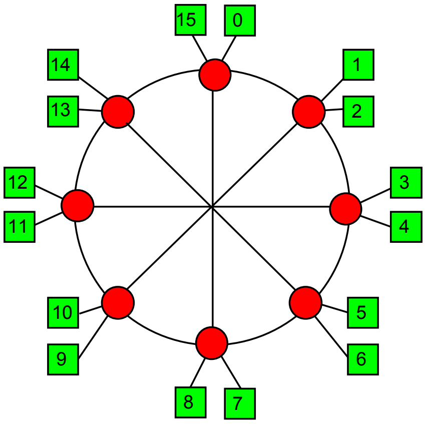

Fig. 4. Fat Octagon topology

Four representative collective communications were designed under the assumption of one-port model, full duplex links and wormhole routing. The results indicated as upper bounds appear in Table 2, together with the lower bounds obtained from equations (6) – (10).

Table 2. M-to-N Broadcast and Scatter, lower and upper bounds, on Fat Octagon topology (P = 16)

Lower bounds	MNB	MNS
8 to the same 8	7	7
8 to other 8	8	10
8 to all 16	8	15
all 16 to all 16	15	15

Upper bounds	MNB	MNS
8 to the same 8	7	10 /7*)
8 to other 8	8	10
8 to all 16	8	15
all 16 to all 16	15	17+)

*) with non-minimum routing
+) non-minimum routing not found

7. Conclusions

The lower bounds $\tau_{CC}(G)$ on number of CC steps were derived for general case of M-to-N collective communications. The application-oriented CCs of this kind are of increasing importance on multiprocessor SoCs. One example is when one group of processors finishes a task and a different size group continues and needs the intermediate results from the first group.

The evolutionary algorithms such as MBOA and HGSA have been used for scheduling CC in the minimum number of steps without creating a conflict (a common link in two transfers in the same step). The (sub)optimal solutions can be obtained for the case of minimum routing, slim or fat nodes and any type of CC. The really obtained upper bounds $\tau^{CC}(G)$ were presented for fat Octagon topology for illustration only. Since a distance-insensitive wormhole switching was assumed, the real communication times can be obtained approximately from the number of start-ups $\tau^{CC}(G)$ plus the serialization delay $m\ t_1$,

$$t_{CC} = \tau^{CC}(G) \times\ t_S +\ m\ t_1, \qquad (12)$$

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 thus be estimated with a good degree of accuracy.

Optimization is not dependent on the size of networks: the algorithms are always the same, only the data structures (specification of networks) differ. The size of solvable problems is limited by excessively increasing computing time even on a 10-blade cluster. This excludes frequent changes in topology of large networks. The future research should investigate scalability limits of the both presented algorithms (now around 64 – 128 nodes) and possible improvements of these algorithms by means of cleverer heuristics for even higher scalability.

It is seen from the results, that the upper bounds are equal or close to lower bounds. Providing multiple CPU ports for simultaneous communication is also of interest in maintaining performance close to theoretical limits.

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] Gabrielyan, E., Hersch, R.D., "Efficient Liquid Schedule Search Strategies for Collective Communications". *Proc. of the 12th IEEE International Conference on Network ICON* 2004, Singapore, Vol. 2, Nov. 2004, pp 760-766.

[5] Ocenasek, J.: *Parallel Estimation of Distribution Algorithms*, PhD. Thesis, Faculty of Information Technology, Brno Univ. 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.

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

Acknowledgement

This research has been carried out under the financial support of the research grant "Architectures of Embedded Systems Networks", GA102/05/0467, Grant Agency of Czech Republic, 2005-2007 and "Security-Oriented Research in Information Technology", MSM 0021630528.