Complete Exchange Algorithms for Meshes and Tori Using a Systematic Approach

Luis Díaz de Cerio, Miguel Valero-García and Antonio González

Computer Architecture Department - Universitat Politècnica de Catalunya (Spain)[1]

Abstract. Frequently, algorithms for a given multicomputer architecture cannot be used (or are not efficient) for a different architecture. The proposed method allows the systematic design of complete exchange algorithms for meshes and tori and it can be extended to some other architectures that may be interesting in the future.

1 Introduction

Complete exchange is a global collective communication operation in which every process sends a unique block of data to every other process in the system. Since complete exchange arises in many important problems, its efficient implementation on current parallel machines is an important research issue.

Multidimensional meshes and tori have received a lot of attention as convenient topologies for interconnecting the nodes of message-passing multicomputers. The fixed-degree property of these topologies makes them very suitable for scalable systems and solving communication intensive problems, such as complete exchange, become a challenge.

Some authors have proposed algorithms for a given scenario that cannot be applied to other scenarios with a reasonable efficiency. It is also frequent that the ideas which inspire a concrete algorithm for a given architecture cannot be used to derive algorithms for others (i.e. the idea of building spanning graphs, which has inspired efficient algorithms for tori, cannot be applied to meshes, where nodes are not symmetric).

The proposed method enables the systematic design of complete exchange algorithms for a wide range of scenarios, including k-port C-dimensional meshes and tori. The method produces efficient algorithms, outperforming in many cases the best known algorithms for a significant range of the system parameters (number of nodes, problem size, communication parameters, etc.).

We have developed analytical models upon a small set of system parameters. These simple models enable a quick and general comparison among different algorithms and are good enough to show the potential goodness of the method.

1. *Author's address:* Computer Architecture Department, Universitat Politècnica de Catalunya, Jordi Girona 1-3, 08034 Barcelona (Spain). *E-mai:* ldiaz@ac.upc.es

2 Considered Scenarios

An scenario is defined by the following features: topology, dimensionality, switching model, port model.

As interconnection topology we consider C-dimensional meshes and tori. The dimensionality of the mesh or torus (C) will take normally the values 1, 2 or 3.

The most frequent switching model is circuit switched. This term includes direct connect, wormhole and virtual cut-through [6]. We model the cost of sending the message in a circuit switched model assuming no conflicts in the use of the system resources, as $t_s + dt_p + Nt_e$, where t_s is the communication start-up, t_p is the time to switch an intermediate node and t_e is the communication time per size unit. A key issue in the design of parallel algorithms for a circuit switched model is the use of conflict-free communication patterns.

The port model defines the number of channels connecting every processor with its local router. In one-port model, every node can send and/or receive only one message at the same time. In all-port model the node can send and/or receive at the same time a message through every external link.

3 The Method

The method (complete description in [2]) starts from a particular parallel algorithm for complete exchange. This algorithm belongs to the CC-cube class of algorithms [1]. CC-cube algorithms are simple and useful to solve many important problems. The original CC-cube algorithm is first transformed in a systematic way through the communication pipelining technique [3], to produce a pipelined CC-cube algorithm. The objective of this transformation is to introduce a certain amount of process level parallelism, which will later enable the exploitation of the machine resources.

The first issue in the mapping of the pipelined CC-cube onto the target scenario is the allocation of the pipelined CC-cube processes to the mesh/torus nodes. The allocation problem can be formulated as an embedding of a hypercube graph onto a mesh/torus graph. We have adopted the standard embedding of hypercubes onto meshes [5] and the xor embedding of hypercubes onto tori [3].

After embedding the pipelined CC-cube onto the mesh/torus, it is necessary to determine how the messages to be exchanged at the end of every iteration of the algorithm must be routed to avoid conflicts. This is the step where the particularities of the target scenario such as circuit switching model and port model play an essential role. The proposed routing algorithms are optimal in terms of communication cost and use a simple dimension-ordered minimal routing policy.

4 A CC-Cube Algorithm for Complete Exchange

In this section we describe the CC-cube algorithm for complete exchange which is used as starting point for the proposed method. Figure 1 shows a particular example in which $d = 3$.

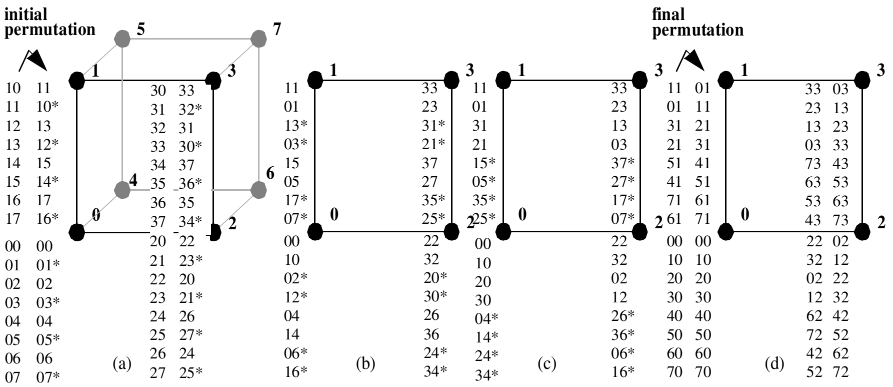

Figure 1 A CC-cube algorithm for complete exchange, with *d*=3.

Every node initially stores 2^d blocks of data in a vector of blocks M. Each block of data will be identified by the pair (m, j), where m is the source node and j is the destination node for the corresponding block. For clarity, figure 1 only shows the blocks of data corresponding to nodes 0, 1, 2 and 3 of the CC-cube. Note that block (m, j) is initially stored in position $M[j]$.

Initially, every node performs a permutation of vector M. After this permutation, block (m,j) is stored in position $M[m \text{ XOR } j]$, in node m (see figure 1.a). The objective of this permutation is to store a block in the location of M given by the binary value obtained by setting all the bits corresponding to the dimensions that the block must traverse (i.e., $M[3]$ of node 1 stores the block (1,2) since this block must traverse dimensions 0 and 1 to reach its destination). Then, in every iteration i of the CC-cube, a subset of 2^{d-1} blocks are extracted from M to build the vector x_i that will be sent to the neighbor in dimension i. In figure 1, the blocks which are selected in every iteration are marked with an asterisk. Because of the initial permutation, all the nodes obtain their corresponding blocks from the same locations of M. In particular, in iteration i a block in position $M[j]$ is selected if the i-th bit of the binary form of index j is set. After the message exchange, the received blocks are stored in the positions of M which were occupied by the sent blocks.

After the three iterations required for complete exchange, a new permutation is required to leave the blocks in their final positions in M. This permutation is exactly the same as the initial one (see figure 1.d).

The above algorithm was proposed in [4] in order to minimize the maximum orbit length. We propose a slight modification of the algorithm in order to meet the requirements of the communication pipelining technique. This modification refers to the order in which the blocks are sent in each iteration. In particular, to build a message x_i, the blocks are always arranged in reverse order with regard to their positions in M. For instance, x_0 in node 0 contains blocks 07, 05, 03 and 01, in this order.

5 Concluding Remarks

In this paper (a strongly reduced version of [2]), a method for the systematic design of complete exchange algorithms for a wide range of scenarios has been proposed. Starting from a particular algorithm for complete exchange, the method uses communication pipelining to introduce node level parallelism, efficient embeddings of hypercube onto mesh/torus to map the pipelined algorithm onto the target scenario, and an efficient message routing to exploit the communication resources of the machine.

Besides its generality, an important feature of the method is the possibility of tuning the algorithm for the particular machine configuration, by choosing an optimal value of the pipelining degree, which is an algorithm parameter. This feature makes possible to obtain a high performance for a wide range of values of the machine and the problem parameters.

Under common analytical modeling assumption, the algorithms obtained through the proposed method outperform previous proposals for a significant range of values for the machine parameters and block size, in almost all the considered scenarios. In many cases the proposed algorithms are about twice as fast as the best previous proposal.

Acknowledgments

This work has been supported by the Ministry of Education and Science of Spain (CICYT TIC-98/0511) and the European Center for Parallelism in Barcelona (CEPBA).

References

[1] L. Díaz de Cerio, A. González and M. Valero-García, Communication Pipelining in Hypercubes, *Parallel Processing Letters*, Vol.6, No.4, December 1996.

[2] L. Díaz de Cerio, M. Valero-García and A. González, A Systematic Approach to Develop Efficient Complete Exchange Algorithms for Meshes and Tori, *Research Report* UPC-DAC-97-29, http://www.ac.upc.es/recerca/reports/INDEX1997DAC.html

[3] A. González, M. Valero-García and L. Díaz de Cerio, Executing Algorithms with Hypercube Topology on Torus Multicomputers, *IEEE Trans. on Parallel and Distributed Systems,* Vol. 6, No. 8, August 1995, pp. 803-814.

[4] S.L. Jonhnsson and C.-T. Ho, Optimal All-to-All Personalized Communication with Minimum Span on Boolean Cubes, in proceedings of the*6th Distributed Memory Computing Conf.*, 1991, pp. 299-304.

[5] S. Matic, Emulation of Hypercube Architecture on Nearest-Neighbor Mesh-Connectec Processing Elements, *IEEE Trans. on Computers*, vol. 39, No. 5, May 1990, pp. 698-700.

[6] J.G. Peters and M. Syska, Circuit-Switched Broadcasting in Torus Networks,*IEEE Trans. on Parallel and Distributed Systems,* Vol. 7, No. 3, March 1996, pp. 246-255.