Design of Group Communication for Regular and Irregular Networks

Miloš Ohlídal[(1)]
Josef Schwarz[(2)]

[(1), (2)]Brno University of Technology, Faculty of Information Technology
Department of Computer Systems, Božetěchova 2, 612 66 Brno, CZ
Tel.: +420-5-41141210, fax: +420-5-41141270, e-mail: {ohlidal, schwarz}@fit.vutbr.cz

Abstract: *Communication between non-adjacent processors in regular and irregular interconnection networks mostly relies on routing tables. Because the tables generally cannot be derived by means of an analytical approach, we have focused on the utilization of the hybrid parallel genetic simulated annealing algorithm HGSA [1] to design optimal or sub-optimal routing schedule for a sequence of communication steps (neighbor-to-neighbor transfers) during the group communications. The efficiency of the HGSA algorithm was tested on the regular hypercube topology (with the known complexity of communication) and on the irregular AMP (A Minimum Path) topology.*

Keywords: *group communications, hybrid parallel genetic simulated annealing, communication architectures, parallel processing*

1. Introduction

Processors in common distributed-memory parallel computers are often interconnected with networks that are node-symmetric, i.e. each node has the same number of channels of the network. Such regular networks like ring, 2D-torus or hyper-cube take advantage of the same relatively simple routing schedule identical for all nodes. All-to-all communications require so that all nodes communicate simultaneously without conflict, i.e. no communication channel between nodes can be used at any time in one direction by more than once. However, there are known applications with irregular networks have hopeful property and therefore this type of networks is often preferred. An example of such irregular network topology is the eight-processor AMP (A Minimum Path) configuration (see Fig.1), designed especially to minimize the network diameter and the average inter-processor distance [2]. The AMP networks for node count p = 5, 8, 12, 13, 14, 32, 36, 40, 53, 64, 128, 256 are known. The *SC* node denotes a system controller (host computer) that sends input data to processing nodes and collects results. Each processing node can communicate simultaneously on four bi-directional full duplex links.

Because the implementation of group communication algorithms is seldom mentioned in literature [3], [4], we first define their general features in the next Chapter. In Chapter 3 the HGSA algorithm utilized for some low-cost group communication is described. Finally in Chapter 4 experimental results were presented. We summarize the contribution of the paper in Chapter 5.

2. Hardware Models and Group Communication

Communications between two nodes/partners (p2p) or among all (or a subset) of partners engaged in parallel processing have a dramatic impact on the speedup of parallel applications. Performance modeling of p2p and group communications is therefore important for the design of application specific systems.

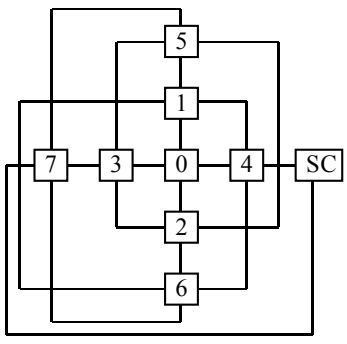

Fig. 1. Eight-processor AMP configuration

The simplest time model of communication in distributed memory systems uses a number of communication steps (rounds): point-to-point (p2p) communication takes one step between adjacent nodes and a number of steps if the nodes are not directly connected. The communication time is composed of a fixed start-up time t_S at the beginning and of a component that is a function of distance d (the number of channels on the route or *hops* a message has to make) and message length m in proper units (words or bytes). For distance-sensitive store-and-forward (SF) switching we have

$$T_{SF} = t_S + d\,(t_r + m\,t_1) \qquad t_r << m\,t_1$$

and for distance-insensitive wormhole (WH) switching

$$t_{WH} = t_S + d\,t_r + m\,t_1$$

where t_r is a routing delay plus the switching delay and inter-router latency and t_1 is the unit-message transfer time. In this linear model of communication we assume no channel contention.

Next, we have to distinguish between unidirectional (simplex) channels and bi-directional (half-duplex, full-duplex) channels. The number of bi-directional channels between the CPU and a router (ports) that can be engaged in communication simultaneously (one-port or all-port models) has also an impact on number of communication steps and communication times, and if nodes can combine/extract partial messages with negligible overhead (combining nodes) or can only re-transmit/consume original messages (non-combining nodes). Finally we have to take into account a switching technique (SF or WH) and network topology.

A single neighbor-to-neighbor (n2n) communication, multiple n2n communications, and a single p2p communication are always deadlock free. For multiple simultaneous p2p communications (including permutation) the deadlock freedom is ensured by the acyclic property of a channel dependency graph. In that graph the nodes correspond to channels and a directed edge connects two nodes if and only if a message coming through the first channel is routed to the second one. A deadlock-free routing table for 8-processor AMP network and the related channel dependency graph are given in Table 1 and Fig. 2 - originally published in [5]. In case of routing a message from node 2 to node 7 via node 6 (instead node 3), as shown by the dotted line, a cycle and a possibility of a deadlock can arise.

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 generating from a subset or the set of all processors are called group or collective communications such as:

- **OAB** (One-to-All Broadcast): One node sends the same message to all other nodes.
- **OAS** (One-to-All Scatter): One node sends distinct messages to all other nodes.
- **AOG** (All-to-One Gather): A reverse operation to OAS – all nodes send a message to a single node.
- **AAB** (All-to-All Broadcast): All nodes perform OAB at the same time, each node sends its message to all other nodes.
- **AAS** (All-to-All Scatter): All nodes perform OAS at the same time, each node sends distinct (private) messages to all other nodes.
- **Permutation**: It is produced sequence of nodes and each node communicates with its neighbor according to this sequence.

Destination

Source \ PE	0	1	2	3	4	5	6	**7**	SC
0	**x**	N	S	W	E	N	S	W	E
1	S	**x**	W	N	E	N	W	W	E
2	N	E	**x**	W	S	E	S	**W**	S
3	E	N	S	**x**	E	N	S	W	W
4	W	N	S	W	**x**	N	S	S	E
5	W	S	E	W	S	**x**	N	N	N
6	N	W	N	S	E	S	**x**	S	E
7	E	N	E	E	S	N	S	**x**	W
SC	W	W	W	S	W	S	S	S	**x**

Table.1. 8-processor AMP routing table

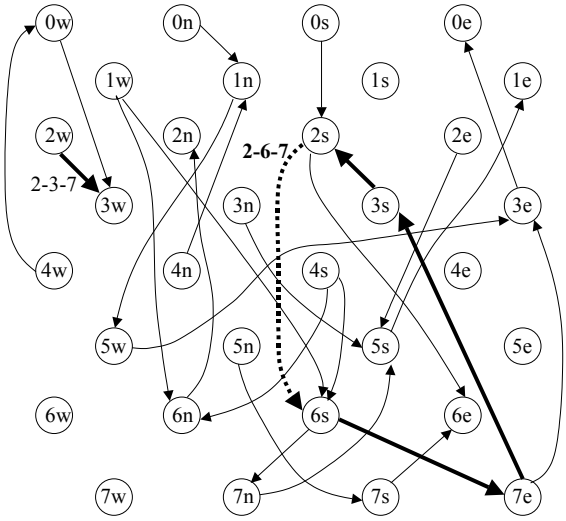

Fig. 2. The channel dependency graph

Implementation of group communications is inherently prone to a deadlock. The acyclic property of the channel dependency graph is not sufficient, it is applied only to the situation when a group of nodes is sending messages to distinct partners (permutation routing or a subset of it). If each node sends or receives messages to or from more than one partner in a loop asynchronously, a fetch deadlock can appear (at least two nodes execute pending send operation and cannot receive). That is why we use a synchronized communication model and we assume that the communication proceeds in synchronized rounds (steps). In SF networks one round (step) consists of a set of parallel (simultaneous) hops of packets between adjacent nodes. In WH networks, a round is a set of conflict-free paths between simultaneously communicating pairs and one round takes time given by the slowest communicating pair.

In the rest of this paper, we focus especially on OAB and AAB operations realized on interconnection networks with full-duplex links, SF switching with non-combining nodes and all-port communication networks. These communication tasks cause the highest communication traffic and their timing overhead greatly depends on capabilities of particular communication hardware. In the group communication is the only way to find optimal or sub-optimal schedule of communication steps using combinatorial optimization, especially in irregular topologies.

3. Structure of the routing algorithm

The routing algorithm consists of two main phases. In the first phase, all paths without cycle among nodes in a particular topology using a BFS algorithm are specified. In the second phase, HSGA algorithm is used to build the optimal schedule based on the paths found in the first phase. The schedule includes n2n communications for a given group of communications (OAB or AAB), especially for irregular networks where the schedule cannot be constructed analytically.

3.1. The path search

This task is performed by the breadth-first search algorithm. A tree is gradually constructed, one level at a time, beginning from a root which is assigned an index of a source node. When a new level of the tree is generated, every node at the lowest level (leaf) is expanded. When a node is expanded, its successors are determined as all its direct neighbors except those, which are already located at higher levels of the tree (it is necessary to avoid cycles). Construction of the tree is finished when a value of at least one leaf is equal to an index of a destination node. Destination leaves' indices determine identity of found paths, which are then stored as sequences of node indices.

When the list of the paths is created, it is used for iterative construction of a time schedule of the given group communication by employing the HGSA algorithm. A chromosome represented by an array of genes encodes the solution. Every gene encodes a time schedule of a single message transmission from a given source node to a destination node. This schedule consists of an identity of a path, which is used for routing the message, and time step at which the communication starts, see Fig. 3. Number of genes G in every chromosome is determined by

the type of communication to be scheduled and by the number of nodes P in a target processor architecture expressed as $G = P\text{-}1$ for OAB or $G = P(P\text{-}1)$ for AAB.

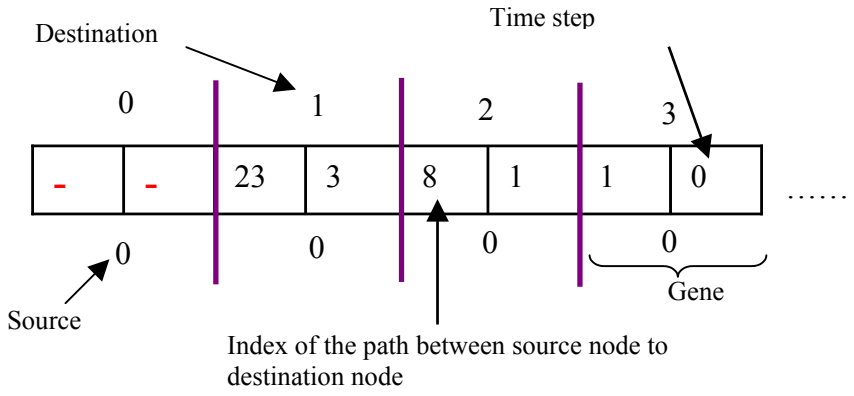

Fig. 3. The structure of the chromosome

3.2. Hybrid parallel genetic simulated annealing (HGSA)

HGSA is a hybrid method utilizing parallel SA with genetic operators. The flow of the algorithm is shown in Fig. 4 originally published in [6]. The parallel SA is built on the base of master-slave configuration of processes. In each process the sequential SA is running.
During the communication which is activated each 10^{th} iteration of Metropolis algorithm, each slave process sends its origin solution to master. Master keeps to oneself one solution and one random chosen solution is sent to each slave. The random choice is based on the roulette wheel, where the biggest probability of selection has the individual with the best fitness function. After the communication all processes have two individuals. From two parent solutions two children solutions are generated using two-point crossover.

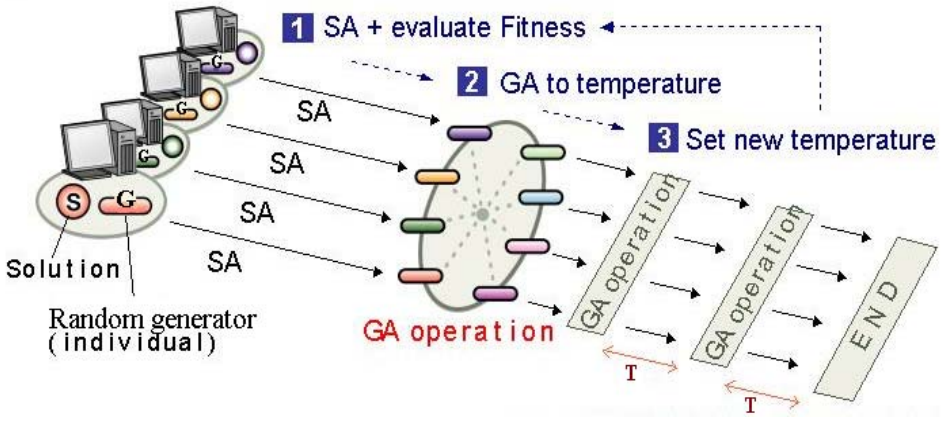

Fig. 4. Hybrid parallel genetic simulated annealing [6]

Solution with the best fitness function is selected and mutated. In case of the parent solutions mutation is always performed. Offspring are mutated by predefined probability. Mutation is performed on randomly selected genes changing the index of a path (if more than one path is available) and the time step.
Consecutively a new solution is selected from the origin solution of each SA process and the solution, which was obtained by genetic operations. The Metropolis criterion is used for the selection process.

The fitness function of an individual is calculated as the sum of the number of conflicts among each pair of genes and the number of steps of designed communication. The number of steps of designed communication is calculated as time step plus length path of all genes. The number of conflicts has bigger priority so it is multiplied by a weight constant. A conflict is defined as follows: two genes represent a *conflict* when they use the same communication channel in the same time step. It is obvious that we concern only with such final

solutions, which have no conflicts, i.e. whose cost is equal to zero. We accept only solutions with zero number of conflicts and consecutively with minimized number of steps of all communication.

3.3. Heuristics

In HGSA two heuristics to speed up the convergence are used. They decrease the probability to get stuck in the local optimum during the optimization. The heuristics are capable to reduce the path length. The first heuristics is used after initialization of chromosome and after selection phase utilizing the Metropolis algorithm. During the initialization a path from source node to destination node and time step are randomly selected. The principle is simple: The path from source node to destination node consists of node sequence. If the destination node is occurred in another path and the number of nodes in sequence is smaller, the origin path is changed according to path where the destination node is occurred, see Fig. 5., e.g. the first sequence 0451 is reduced to the sequence 01 because the searched destination node 1 is on the second position in the second sequence. It means that the number of nodes between source node and destination node in the second sequence is smaller than in the first sequence.

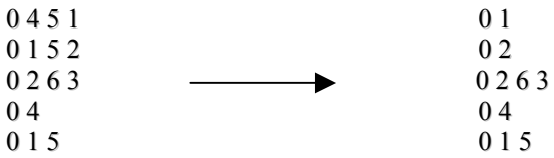

Fig. 5. Illustration of the first heuristic

The second heuristic is used in the case of zero number of conflicts between genes of chromosome. This heuristic performs the analysis of the chromosome. It searches the longest path in some gene in the chromosome. If the number of the longest paths doesn't exceed some defined value, it tries to reduce the size of these longest paths. It applies the mutation on these longest paths. In case the size of the path is reduced the new gene replaces this origin one.

4. Experimental results

HGSA has already been implemented in C using MPI [7] routines for message passing and it can therefore be compiled and run on any architecture (clusters of workstations, MPPs, SMPs, etc.) for which an implementation of MPI standard is available.

Scheduling of OAB and AAB communication for two different architectures - hypercube and AMP was tested. The number of processors in the target architecture varied from 8 to 128 in case of OAB. The architecture with 8 and 16 processors was only tested in case of AAB. Hypercube was chosen because of its regular topology with known optimal scheduling so that it can serve as a convenient benchmark. OAB communication complexity measured by the number of time steps in the schedule found by HGSA so far, are shown in table 2. The first column specifies the node count in the target architecture, the second column shows the number of steps achieved for a hypercube and the third column shows the number of steps achieved for AMP architecture. For predefined number of nodes/processors and for the both architectures ten runs was applied, see the second column of table 3. We counted only the number of global successes, which was obtained during the execution. The success level in percent rate is shown in the third column for AMP and in the fourth column for hypercube.

	# steps	
# nodes	**hypercube**	**AMP**
8	3	2
16	4	-
23	-	3
32	5	3
42	-	4
53	-	3
64	6	-
128	7	-

Table 2. Results of OAB optimisation

		Success rate [%]	
# nodes	**# experiments**	**AMP**	**Hypercube**
8	10	100	100
16	10	-	100
23	10	100	-
32	10	100	100
42	10	100	-
53	10	100	-
64	10	-	100
128	10	-	100

Table 3. Success rate of the global optimum achievement

Scheduling of AAB communication was tested only for hypercube and AMP architectures with only 8 and 16 nodes/processors. We obtained optimal solution with 3 steps for 8-nodes and 4 steps for 16-nodes. But computation time is too long – for 16 processors it is 1hour and 20 minutes. To reduce the time complexity it is necessary to implement more efficient heuristics.

5. Conclusions

We have implemented the HGSA algorithm for the optimization of OAB and AAB group communications. The ability of HGSA algorithm to find optimal or suboptimal solutions was confirmed in the case of the hypercube architecture. In case of the OAB communication the global optimum was found for up to 128 processors. In case of the AAB communication the global solution was obtained for an eight and sixteen nodes network. The main advantage of the proposed approach is the ability to schedule the group communications for various network topologies (e. g. AMP) with unknown optimal (minimal) number of steps. It is important especially for the irregular topologies where the analytical approach cannot be used.
The future research will be focused on the design and implementation of more efficient heuristics for the AAB communication in particular for complex network topologies.

Acknowledgement

This research has been partly carried out under the financial support of the research grant GA 102/02/0503 "Parallel system performance prediction and tuning" (Grant Agency of Czech Republic).

References

[1] Ohlídal, M., Schwarz, J.: Hybrid parallel simulated annealing using genetic operations, Brno, 2004

[2] Chalmers, A.-Tidmus, J.: Practical Parallel Processing. International Thomson Computer Press, 1996.

[3] Gabrielyan, E. - Hersch, R.D.: Network′s Liquid Throughput: Topology Aware Optimization of Collective Communication. Unpublished work, 2003.

[4] Defago, X., Schiper, A., Urban, P.: Total Order Broadcast and Multicast Algorithms: Taxonomy and Survey, technical report DSC/2000/036, 2003

[5] Staroba, J., Dvořák, V.: Design of Low-Cost Communication Algorithms for Irregular Networks, Université de Haute Alsace, Colmar, 2004

[6] Miky, M., Hiroyasu, T., Wako, J., Yoshida, T.: Adaptive Temperature Schedule Determined by Genetic Algorithm for Parallel Simulated Annealing, Doshisha University, Kyoto, 2003

[7] Manual MPI Document reasonable on URL:
http://bitscap.bits-pilani.ac.in/param/public_html/mpi_basic_calls.html, (May 2002)