An Evolutionary Design Technique for Collective Communications on Optimal Diameter-Degree Networks

Jiri Jaros
Brno University of Technology
Bozetechova 2
612 66, Brno, CZ
+420 54114 1207
jarosjir@fit.vutbr.cz

Vaclav Dvorak
Brno University of Technology
Bozetechova 2
612 66, Brno, CZ
+420 54114 1149
dvorak@fit.vutbr.cz

ABSTRACT

Scheduling collective communications (CC) in networks based on optimal graphs and digraphs has been done with the use of the evolutionary techniques. Inter-node communication patterns scheduled in the minimum number of time slots have been obtained. Numerical values of communication times derived for illustration can be used to estimate speedup of typical applications that use CC frequently. The results show that evolutionary techniques often lead to ultimate scheduling of CC that reaches theoretical bounds on the number of steps. Analysis of fault tolerance by the same techniques revealed graceful CC performance degradation for a single link fault. Once the faulty link is located, CC can be re-scheduled during a recovery period.

Categories and Subject Descriptors

I.2.8 [**Artificial intelligence**]: Problem Solving, Control Methods and Search – *heuristic methods, scheduling*.

General Terms

Algorithms, Performance, Design.

Keywords

Collective communications, communication scheduling, evolutionary design, interconnection networks.

1. INTRODUCTION

High performance computing platforms have been recently dominated by clusters of multi-core processor nodes [1] or by many cores interconnected by a NoC (Network on Chip). The basic requirement for building the HPC systems turned out to be the low power consumption, in order that system parts could be close together and communication time thus minimized. For the same reason the CPU cores should be simple and processing nodes should be interconnected directly, without intermediate switches and routers. A class of interconnection networks of interest in this paper covers therefore direct networks, which for performance-driven environments converge on the use of pipelined cut-through (CT)/wormhole (WH) message transmission and source-based routing algorithms.

In this paper, we want to analyze the complexity of collective communications in a class of networks whose size is equal or close to upper bounds known for the given node degree and diameter. Simply said, as many nodes as possible are connected by a regular network with a uniform node degree d (a d-regular network), with inter-node distance up to D. Systems of this sort are more compact than others and can support faster communications, too. As far as the authors know, performance of collective communications on such networks has not been studied as yet. The reason may be that, until recently [2], these networks have not been used in commercial systems. Contribution of the paper is in assessment whether the theoretical lower bounds of CC times are reachable at all or how close we can get to them. This is shown directly by designing (evolving) initial invalid schedules up to the optimal or sub-optimal valid (conflict-free) variants.

Evolutionary techniques applied already to CC scheduling problem on hypercubes of medium size (tens of nodes) [3] were able to find optimum solutions obtained by mathematical means. However, for networks studied in this paper no analytic methods for scheduling exist. The results can be compared to theoretical lower bounds only. The paper is structured as follows. In Section 2, Moore networks and networks close to them in size are defined, based on underlying graphs and digraphs. Section 3 specifies the scheduling problem for CC and presents an improved evolutionary algorithm for its solution. The results of CC scheduling in various network topologies are summarized and discussed in Section 4. Section 5 deals with fault tolerance of interconnection networks and possible recovery from a faulty link. Results obtained by evolutionary approach are discussed in Conclusion and possible future improvements are suggested.

2. OPTIMAL DIAMETER-DEGREE NETWORKS

Pair-wise (point-to-point) as well as collective (group) communications involving all processors are frequently used in parallel processing and their timing complexity has a dramatic impact on performance. Since processors are connected only sparsely, the message can reach a destination processor directly, if source and destination processors are neighbors, or else through some intermediate nodes. The communication time from issuing the send request by one CPU until receiving data by another CPU represents an overhead of parallel processing which has to be minimized. The pipelined message transmission is considered only little sensitive to the source-destination distance; however, accumulating delays when traversing several nodes on the way should be minimized as well. Therefore the networks of diameter D connecting the maximum number of nodes N of the given degree d are of interest [4].

The upper bounds on the number of P nodes with degree $d > 2$ that can be connected into an undirected graph, shorty graphs (1), or directed graph, shortly digraphs (2), of diameter $D \geq 1$ are known as Moore bounds [4]:

$$P \leq 1 + d + d(d-1) + ... + d(d-1)^{D-1} = \frac{d(d-1)^D - 2}{d-2} \quad (1)$$

$$P \leq 1 + d + d^2 + ... + d^D = \frac{d^{D+1} - 1}{d-1} \quad (2)$$

A regular graph of degree d and diameter D whose number of vertices equals the above upper bound (1) is Moore graph. If we exclude fully connected graphs with $D = 1$, there exist only a few such graphs:

D=2: $d = 3$, $P = 10$ (Petersen graph)

D=2: $d = 7$, $P = 50$ (Hoffman-Singleton graph)

and no others with the possible exception $d = 57$ (which is still undecided). There are no Moore graphs with $D \geq 3$ and no Moore digraphs either (disregarding trivial cases $D = 1$ or $d = 1$).

Whereas all above Moore graphs have length of the shortest cycle (girth) five, the even girth (6, 8 and 12 only) can also be considered, what leads to the worse upper bound

$$P \leq 2\sum_{i=0}^{D-1}(d-1)^i \quad (3)$$

and to generalized Moore graphs. Few examples follow:

D= 2: $d = 3$, $P = 6$ (utility graph)

D= 3: $d = 3$, $P = 14$ (Heawood graph)

D= 4: $d = 3$, $P = 30$ (Levi graph).

It is an open problem if there are infinitely many generalized Moore graphs of each degree.

Since there are not many known Moore graphs, it is of great interest to find graphs which for a given diameter D and maximum degree d have a number of vertices as close as possible to the Moore bound. The largest known graphs and digraphs are listed in [4] and [5]. The more systematic approach has been used to design "almost" Moore graphs that miss the upper bound by a small number [4] or whose number of nodes approximates the upper bounds asymptotically. The best networks in the latter case are based on Kautz digraphs with $P = d^D + d^{D-1}$ nodes. Table 1 gives all three upper bounds (1) – (3), the largest known graphs [4], Kautz digraphs and generalized Moore graphs, all with degree $d = 3$. It turns out that the largest known digraphs are Kautz digraphs and that digraphs (2) are potentially much larger than graphs (1).

In the following sections we will try to derive communication schedules on the above graph topologies. Since there have been no optimum schedules published, we can make comparison only with the known theoretical lower bounds on communication complexity. These bounds will serve us as a target we want to come to as near as possible.

Table 1. The size of graphs and digraphs with degree d = 3 and diameter D

D	2	3	4	5	6	7
(1)	10	22	46	94	190	382
(2)	**13**	**40**	**121**	**364**	**1093**	**3280**
(3)	6	14	30	62	126	254
[4]	10	20	38	70	132	192
Kautz	**12**	**36**	**108**	**324**	**972**	**2916**
Moore	10	14*	30*	-	-	-

* denotes a generalized Moore graph; digraphs are in bold

3. CC SCHEDULING PROBLEM

In this section we are going to analyze only the frequently used collective communications involving all processors: one-to-all broadcast (OAB), all-to-all broadcast (AAB), one-to-all scatter (OAS, a private message to each partner), and all-to-all scatter (AAS). Some other CCs, like all-to-one gather (AOG), have the same complexity as the basic four types.

Each CC can be seen as a set of point-to-point communications. The CC scheduling problem can be simply described as partitioning this set into as few subsets as possible that follow one another in a sequence of synchronized steps; all communications in one subset proceed in parallel. The main goal is to avoid any conflicts in shared resources – links (channels). Several messages between source-destination pairs can proceed concurrently and can be combined into a single subset if their paths are link-disjoint. 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.

The number k of bi-directional channels between the CPU and a router (ports), that can be engaged in communication simultaneously, has a decisive impact on the number of communication steps; 1-port (k=1) or all-port (k=d) models are most common, see Fig. 1. For the highest performance we will consider only the all-port model (k=d).

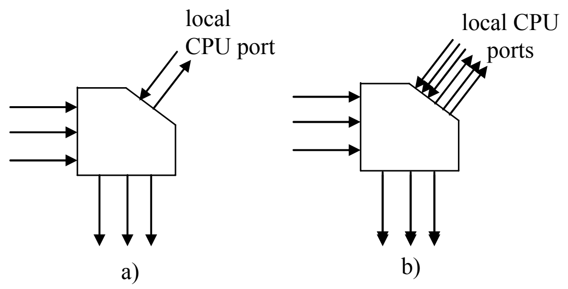

Figure 1. Port models for 3-regular networks a) one-port router b) all-port router

Regardless the graph topology, there are known theoretical lower bounds on the number of communication steps. The broadcast communication (OAB) in WH network cannot be done in less than s steps, where $s = \lceil \log_{d+1} P \rceil$ is given by the number of nodes informed in each step, that is initially 1, $1+1\times d$ after the first step, $(d+1)+(d+1)\times d = (d+1)^2$ after the second step, etc.,…, and $(d+1)^s \geq P$ nodes after step s.

In case of AAB communication, since each node has to accept $P-1$ distinct messages, the lower bound is $\lceil (P-1)/d \rceil$. A similar bound is applied to OAS communication, because each node can inject into the network not more than d messages in one step. $P-1$ pair-wise communications, d of them per step, must be packed into the lowest number of steps in such a way that paths traversed in the optimum broadcast tree are edge-disjoint in each step.

For AAS communication pattern each of P processor sends an individual message to each of P-1 partners. A lower bound for AAS can be obtained considering that one half of messages from each processor cross the bisection and the other half do not. There will be altogether 2 $(P/2)(P/2)$ of such messages in both ways and up to B_C messages in one step, where B_C is the network bisection width [6]. In case of digraphs (graphs), B_C is taken as the (double) the number of (un)directed edges crossed by the bisection. However, some Δ messages originating and terminating in either half of a network cross the bisection as well. This gives the bound $(P^2/2 + 2\Delta)/B_C$ communication steps, since Δ messages cross the bisection twice. Another bound that concerns AAS used to be applied to SF [14] routing only. If Σ denotes the sum of all shortest paths in a graph (from any source to any destination node) and if we can utilize only Pd channels in one step to avoid conflicts, then we cannot schedule AAS in less than $\lceil \Sigma/Pd \rceil$ steps. We have found that for the considered class of networks this latter bound is stronger, even for WH routing.

Table 2 summarizes the lower bounds for general graphs and numerical values for three 3-regular graphs (Petersen P10, Kautz K12 and Heawood H14). Of course, communication bounds for AAB and AAS cannot be ever shorter than those ones for OAB and OAS respectively, if it applies.

Table 2. Lower bounds on complexity of CC in *d*-regular networks with P nodes

CC	CT (WH)	P10	K12	H14
OAB	$\lceil \log_{d+1} P \rceil = \lceil (\log P)/\log(d+1) \rceil$	2	2	2
AAB	$\lceil (P-1)/d \rceil$	3	4	5
OAS	$\lceil (P-1)/d \rceil$	3	4	5
AAS	$\max[\lceil (P^2/2+2\Delta)/B_C) \rceil, \lceil \Sigma/(Pd) \rceil]$	5	7	8

4. CC SCHEDULING ALGORITHM

The selection of Evolutionary Algorithms (EA) for the scheduling problem has been justified already in [3]. Although a new methodology of designing near-optimal CC schedules is independent of the particular evolutionary algorithm, we restricted ourselves only to a simple EDA evolutionary algorithm without gene dependencies (UMDA) in this work.

Univariate Marginal Distribution Algorithm (UMDA) [9] is a very simple EDA [12] (Estimation of Distribution Algorithm) which does not reflect any interaction between genes (variables/solution parameters). The main advantages of this algorithm are better mixing of genetic material than is possible in standard GA [13], very simple implementation and much faster execution than more complex EDAs like BOA (Bayesian Optimization Algorithm [12]) algorithm. Of course, any other EA can be employed. Basic comparison of a success rate and execution time of other types of EA applied to CC scheduling problem can be found in [10, 11].

This section describes, in more details, the elements of our evolutionary approach. Section 4.1 shows the global data structure and a preprocessing phase. Section 4.2 describes how the dataset is encoded, Section 4.3 presents the evaluation function used in EA and Section 4.4 briefly describes acceleration and restoration heuristics used to increase a success rate and reduce execution time required to reach a good result. Parameters of used EA (UMDA) are outlined in Section 4.5.

4.1 Preprocessing Phase

An input data structure maintains a topology description, a definition of CC and a set of senders and receivers. The topology de description is saved in the form of a neighbors list for each node, where the nodes are considered to be neighbors only if they are connected by a simple direct link.

After an input file is loaded, the data have to be preprocessed. The preprocessor takes the topology description and finds all paths (shortest ones in the case of minimal routing) between all source-destination node pairs and stores them into a special data structure. This task is performed by a modified well known Dijkstra's algorithm.

4.2 Encoding

As broadcast and scatter CCs are completely different communication services, candidate solutions are encoded in separate ways.

An optimal OAS schedule designed for 8-node bidirectional ring is shown in Fig. 2. This schedule reaches the lower bound of 4 steps. The initiator, node no. 0, informs two other nodes in each of the first three steps by means of some of the shortest paths found in the preprocessing phase. The last node is informed during the fourth step via one of two possible paths.

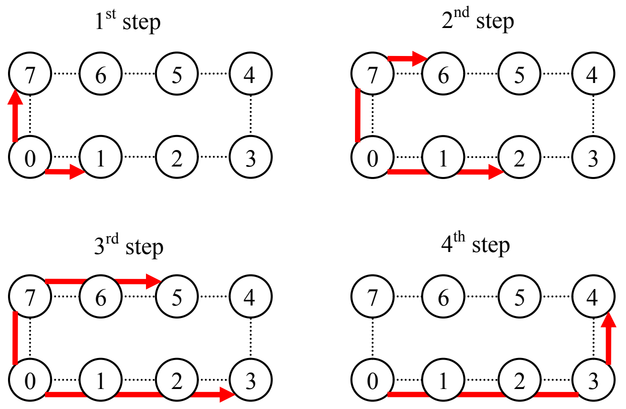

Figure 2. An OAS schedule reaching the lower bound on number of communication steps.

A direct encoding has been designed for OAS/AAS chromosome; i.e. a chromosome contains an exact description of a schedule. The OAS chromosome corresponding to Fig. 2 is displayed in Fig. 3. The chromosome contains *P* genes; each one represents a particular point-to-point communication between the initiator and a destination node. A gene consists of two items: a utilized path (the first component) and the used time step (the second component).

An AAS chromosome is created by extending the vector to a matrix, each row of which corresponds to one of OAS communications.

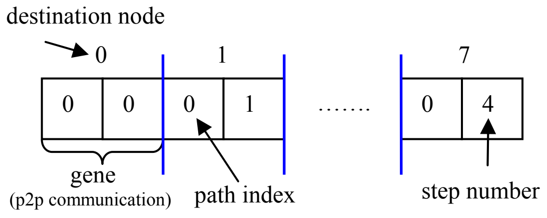

Figure 3. The structure of OAS chromosome for 8-node bidirectional ring

An optimal OAB schedule designed for 8-node bidirectional ring is shown in Fig. 4. This schedule reaches the lower bound of 2 steps. The initiator, node no. 0, informs nodes no. 3 and 6 in the first step (solid arrows). Since the distributed messages are the same for all nodes, these three nodes can become initiators for the second step, such nodes no. 7 and 5 receive the message from the node no. 6, nodes no. 2 and 4 form the node no. 3, and finally node no. 1 from the node no. 0.

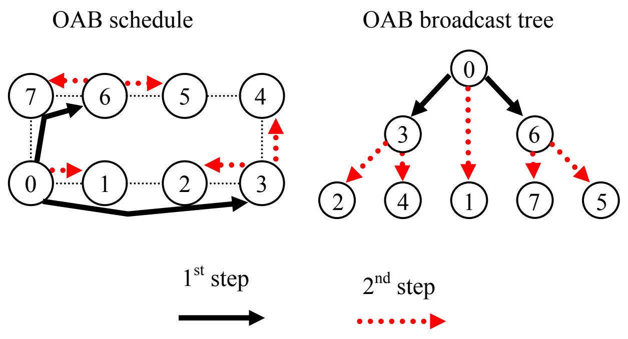

Figure 4. An OAB schedule reaching the lower bound on number of communication steps

An indirect encoding has been designed for OAB; a chromosome does not include a broadcast tree, but only instructions how to create it. Each chromosome consists of *P* genes, one for each destination node, see Fig. 5. Individual genes are composed of three items: a source node index for this destination, the index of the used path, and a step number.

The main disadvantage of this encoding is possible formation of some inadmissible solutions during the process of genetic manipulation. We say that a solution is inadmissible if it cannot lead to a correct broadcast tree. E.g. the situation when in a certain step a node should receive a message from a node that has not received it yet (e.g. node 2 from node 1 in the first step). That is why admissibility has to be verified for each chromosome before evaluating fitness and if it is necessary, the chromosome is restored. The AAB chromosome is then a collection of *P* OAB chromosomes, a kind of a matrix chromosome.

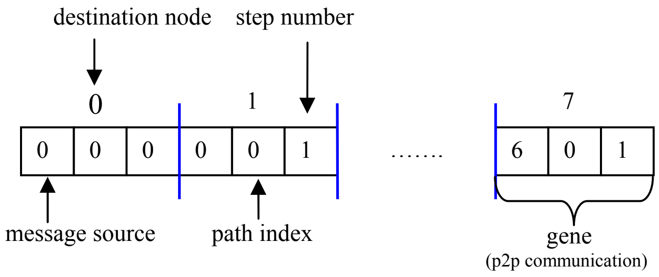

Figure 5. The structure of OAB chromosome for 8-node bidirectional ring

4.3 The Conflict Counting Fitness Function

The main idea of fitness function is based on testing a conflict-free condition. We say two communications are in conflict if and only if they share the same channel in the same communication step (see Fig. 6). The fitness function is based on counting conflicts between all point-to-point communications realized in the same steps. The valid communication schedule for a given number of communication steps must be conflict-free. Valid schedules are either optimal (the number of steps equals the lower bound) or suboptimal. Evolution of a valid schedule for the given number of steps is finished up as soon as fitness (number of conflicts) drops to zero. If it does not do so in a reasonable time, the prescribed number of steps must be increased.

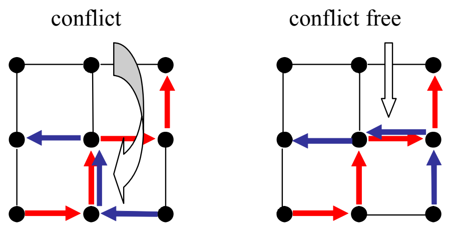

Figure 6. Two point-to-point communications

4.4 Acceleration and Restoration Heuristics

New heuristics have been developed to improve OAS/AAS optimization speed taking into account a search space restriction due to a limited message injection capability of network nodes. Because no node can send more than *k* messages in one communication step (*k*-port model), an acceleration heuristic checks this condition in the whole chromosome and redesigns port's utilization in all communication steps before the fitness function is evaluated.

The second OAS/AAS heuristic replaces the mutation operator in an employed EA. It randomly swaps time slots of two point-to-point communications. These simple heuristics dramatically decrease the initial conflict count and lead to the better convergence of EA.

New heuristics for OAB/AAB chromosome restoration have been also developed and employed. The restoration (a repair of the broadcast tree) proceeds in subsequent communication steps. A check is made for every node whether the node receives the message really from the node already informed. If not so, the source node of this point-to-point communication is randomly replaced by a node that has already received the message. A change of the source node has naturally an impact on utilized channels. Hence the original path is replaced by newly chosen one from a list of exploitable paths between new source-destination pair.

To accelerate the convergence of the EA, an OAB/AAB specific heuristics have been developed. The first optimization injects good building blocks into the initial population. For all point-to-point communications of OAB, the time slot is set initially to the same value (step no. 0). By selecting correct time slots, the restoration heuristic produces corrected broadcast trees that violate the conflict-free condition in much fewer cases.

The second heuristic is based on the search space pruning, and incorporated into the restoration heuristic. If for a given topology, this formula for lower bounds #(AAB steps) > #(OAB steps) is valid, AAB can be performed as a controlled flood; all processors send their message only to uniformed neighbors. In each steps, messages are propagated in waves through the interconnection network. This feature of the interconnection network and AAB communication can be employed with advantage for pruning of the search space. The set of possible receivers of broadcast message in a step can be restricted only to nodes within a given radius $r \in \langle 1, D/2 \rangle$. This restriction leads to a massive reduction of possible engaged shortest paths (alleles for the second gene's component). The suitable radius value is chosen according to a character of the interconnection network; for symmetric interconnection networks r=1. Generally, the lower values of radius lead to faster convergence, but in some cases it is necessary to choose larger values (D/2 in the case of OAB communication) to ensure retrieval a purposeful schedule.

As a consequence of the reception restriction, the restoration heuristic has to be modified. In a process of building the broadcast tree, it can happen that there is no node that can inform a selected node in a given communication step. In this case, the communication has to be postponed to the later time slot, where at least one possible source of broadcast message already exists. In some cases this postponement can cause a number of communication steps is in excess of the requested maximum. This situation is handled by a penalization function. The amount of penalty is given by the sum of all point-to-point communications running over the prescribed maximum number of steps. Finally, this value is added to the conflict count computed by the fitness function.

4.5 Parameters of EA

The simple UMDA evolutionary algorithm has been used for the search for near optimal communication schedules. The value of the population size was set to 60 individuals because higher values did not improve the quality of founded schedules and did not justify an increased computation time. The binary tournament selects the better half of the current population to form the parent subpopulation. The univariate marginal probabilistic model is created according to the parent subpopulation in each generation. New chromosomes are generated by the sampling of the estimated probabilistic model. Each chromosome is mutated by a simple mutation operator with probability of 90%. This operator is responsible for testing and changing possible source-destination paths for particular point-to-point communications. The mutation rate is very high due to great number of source-destination pairs (thousands) whose amount growth exponentially with network diameter D. Finally, the newly generated solutions replace the worse half of the current population.

5. RESULTS OF EVOLUTIONARY OPTIMIZATION

The evolutionary algorithm described previously has been applied to several networks that either already found the commercial application (such as scalable Kautz networks, [2]) or are potential candidates e.g. for NoCs (like non-scalable Petersen (P=10) or Heawood (P=14) networks, Fig. 7).

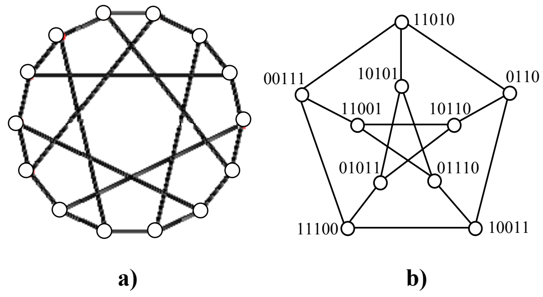

Figure 7. Heawood a) and Petersen graph b)

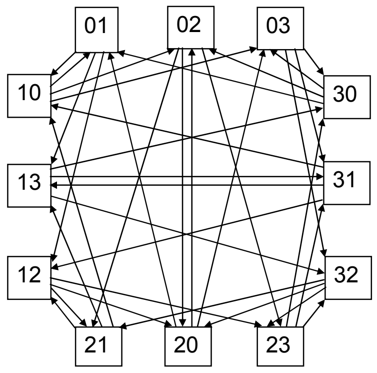

Figure 8. Kautz network with *d* = 3 and *D* = 2

As for the 12-node Kautz network (Fig. 8), number of steps in all four CCs is equal to the lower bound and cannot be improved any more. The resulting schedules are presented only for the most complex all-to-all communication patterns in Table 3 and 4. Some empty slots in Table 3 show that not all links are used in every step of AAS. On the other hand, the lower bound for AAB is very tight (12/3=4 steps) and indicates that all the links are busy in all 4 steps.

Let us note that the presented solution is not unique, several solutions have been found both for AAS and AAB patterns.

Table 3. AAS in 7 steps on Kautz12 network

steps →

src	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	9	2,4,5	3,7,A	6,B	1		8
1	7,9	2,A	0,5,8	3	6	4,B	
2	7	1,3,8	4,9	0	5,B	6,A	
3		1	0,B	8	5,9	2,4,7	6,A
4	A	0,6	1,8		3,B	2,5,9	7
5	4,B	0	A	7,8	2	6,9	1,3
6	2,5	0,4	1	A	8,9	3,7	B
7	0,1,A	6		3,9	8	5,B	2,4
8		2	9,B	1,7,A	4	3	0,5,6
9	6,A	7	2,4	5,8	0,3,B	1	
A	1,7,B	8	3,6	2,5,9		4	0
B	5,6		0,2	A	4,7	1,3	8,9

Table 4. AAB on Kautz12 network in 4 steps (x : message does not move)

src id /msg	three subtrees broadcasting source message		
01=0	0-3-1 0-3-2	0-4-x-x-9 0-4-x-x-A 0-4-x-x-B	0-5-x-6 0-5-x-x-7 0-5-x-x-8
02=1	1-6-3 1-6-4 1-6-x-x-5	1-8-A 1-8-x-9 1-8-x-B	1-x-7-0 1-x-7-x-2
03=2	2-9-8 2-9-x-6 2-9-x-7	2-A-5 2-A-x-4 2-A-x-x-3	2-B-1 2-B-x-0
10=3	3-0-4 3-0-5	3-1-8 3-1-x-6 3-1-x-7	3-2-9 3-2-A 3-2-B
13=4	4-9-6 4-9-7 4-9-x-8	4-A-x-3 4-A-x-x-5	4-B-x-x-0 4-B-x-x-1 4-B-x-x-2
12=5	5-7-2 5-7-x-x-0 5-7-x-x-1	5-x-6-x-3 5-x-6-x-4	5-x-8-A 5-x-8-x-9 5-x-8-x-B
21=6	6-3-x-0 6-3-x-1 6-3-x-2	6-4-9 6-4-x-A 6-4-x-B	6-x-5-7 6-x-5-8
20=7	7-0-x-4 7-0-x-5 7-0-x-x-3	7-2-x-9 7-2-x-A 7-2-x-B	7-x-1-x-6 7-x-1-x-8
23=8	8-9-x-x-6 8-9-x-x-7	8-A-3-x-0 8-A-x-5 8-A-x-x-4	8-x-B-1 8-x-B-2
32=9	9-6-x-3 9-6-x-4 9-6-x-5	9-7-0 9-7-x-1 9-7-x-2-B	9-8-x-x-A
31=A	A-3-0 A-3-x-x-1 A-3-x-x-2	A-4-B A-4-x-9-8	A-5-7 A-5-x-x-6
30=B	B-1-6 B-1-x-8 B-1-x-x-7	B-x-0-3 B-x-0-x-4 B-x-0-x-5	B-x-2-x-9 B-x-2-x-A

As the Kautz network is known for its fault tolerance, we have also tested performance degradation under a single link fault. A fault diameter of the Kautz12 network is *D*+2, meaning that among multiple links between any two nodes the longest path is 4. The network performance under a single link fault is given in Table 5 (with node 01 as the source node for OAB and OAS), but the network could operate even under a double link fault. In any case, when the link fault is detected, the new schedule could be computed in 20 seconds on a single processor and then the cluster could continue with a lower performance.

Table 5. Performance of Kautz12 network with a single faulty link (in # steps). A reduced performance is in bold.

Link	OAB	AAB	OAS	AAS
No fault	2	4	4	7
01-10	**3**	**6**	**6**	**9**
01-12	**3**	**6**	**6**	**9**
01-13	**3**	**6**	**6**	**9**
02-20	2	**6**	4	**9**
02-21	2	**6**	4	**9**
02-23	2	**6**	4	**9**
03-30	2	**6**	4	**9**
03-31	2	**6**	4	**9**
03-32	2	**6**	4	**9**
10-01	2	**6**	4	**9**
10-02	2	**6**	**5**	**9**
10-03	2	**6**	**5**	**9**
12-20	2	**6**	4	**9**
All other	2	**6**	4	**9**

Other network topologies investigated in this study have been Octagon [8] and 16-gon with *D*=2 and 3 and with *d*=3. The results are summarized in Table 6, together with uni- and bi-directional rings and a hypercube for comparison. Two integers in one cell separated by a slash indicate that the lower bound (a smaller integer) has not been reached. A single integer represents both the lower and the upper identical bounds reached by EA. An asterisk (*) indicates the fact that a non-minimum routing has been used; otherwise the minimum routing is used everywhere else.

Table 6. Performance of selected networks (in steps)

all-port model	*d*	OAB	AAB	OAS	AAS
Ring 8	1	3	7	7	16
Ring 8	2	2	4	4	8
Octagon 8	3	2	3	3	4
Petersen 10	3	2	3	3	5
Kautz 12	3	2	4	4	7
Heawood 14	3	2	5	5	9/10
16-gon	3	2	5	5	13/17
Levi 30	3	3	10	10	28/31
Hypercube 32	5	2	7	7	16
Kautz 36	3	3	12	12 *	31/34

In the simplest linear time model of CT (WH) communication in distributed memory systems, the real CC times can be obtained as a sum of communication steps, each step composed of a start-up delay plus the serialization delay $m\ t_1$

$$t_{CC} = (t_0 + mt_1) \times \#\,\text{steps}, \tag{4}$$

where m is a message length (in bytes) and t_1 per byte transfer time. Start-up latency t_0 is the sw- and hw-based latency in the source and destination nodes for initializing the cache-to-cache or memory-to-memory DMA transfer and includes possible synchronization overhead. The hardware overhead in routers along the traversed path has been neglected in (4). Contention for links is associated delays are completely avoided in our schedules.

For example a duration of one communication step in CC for typical cluster parameters [7] t_0=1 μs, t_1= 0.5 ns/byte and the message size 1024 byte has the value of 1.512 μs and the resulting CC times range from 4.54 μs (3 steps) up to 55μs (34 steps). According to frequency of CCs and an amount of interleaved computation in a certain application, efficiency of parallel processing can be estimated.

Table 7 shows average execution times of the EA during 5 successful runs. For OAB communication, the values are less than one second for simple network topologies. The longest execution time (hypercube-32) is about 41 seconds. OAS communication is relatively easy; a solution takes always less than one second. On the other hand, a suitable solution for all-to-all communication takes much longer time, especially for AAS communication. An exponential increase of the execution time with network can be observed.

All experiments were realized in sequential manner on IBM Blade servers equipped with 2x dualcore AMD Opteron 275 processors and 4GB RAM.

Table 7. Execution times of EA in seconds, minutes, hours and days (average values during 5 successful runs)

all-port model	*d*	OAB	AAB	OAS	AAS
Ring 8	1	<1s	<1s	<1s	5m6s
Ring 8	2	<1s	<1s	<1s	57s
Octagon 8	3	<1s	<1s	<1s	2s
Petersen 10	3	<1s	2s	<1s	12s
Kautz 12	3	<1s	3s	<1s	23s
Heawood 14	3	<1s	5s	<1s	9m17s
16-gon	3	3s	1m41s	<1s	22m36s
Levi 30	3	3s	2h2m	<1s	1d6h
Hypercube 32	5	41s	28m38s	<1s	4d5h
Kautz 36	3	20s	9h50m	<1s	3d5h

6. CONCLUSIONS

It is seen from the results, that for the networks of interest in this paper, the obtained upper bounds are mostly close or equal to theoretical lower bounds. The only exception is AAS communication in larger networks, where the lower bounds are apparently too tight. In fact, the obtained numerical results have led us to an improvement of theoretical lower bounds for AAS communication. The lower bound for AAS and WH networks in wide use has been [6], [14]

$$\#(\text{AAS steps}) = \left\lceil \frac{P^2}{2B_C} \right\rceil \tag{5}$$

and we have improved it to

$$\#(\text{AAS steps}) = \max\left(\left\lceil \frac{P^2/(2+2\Delta)}{B_C} \right\rceil, \left\lceil \frac{\sum}{Pd} \right\rceil \right) \tag{6}$$

For orthogonal topologies such as hypercubes, the correction is not needed as Δ=0 and two expressions the maximum is sought of are equivalent. However, for other networks is the inclusion of the correction essential and may change the bound dramatically. E.g. for Kautz12 network the lower bound has changed from 3 to 7 steps. The following Table 8 gives the old and new values of lower bounds, as well as all relevant parameters of the networks. (The number of nodes is given at the name of the network). The correction in bold digits has a dominant impact on the bound. If both corrections are in bold, they have the same influence.

Table 8. Old (5) and new (6) lower bounds on the number of AAS communication steps

all-port model	*d*	B_c	2Δ	Σ	(5)	(6)
Ring 8	1	2	0	128	16	16
Ring 8	2	4	0	128	8	8
Octagon 8	3	12	**8**	**88**	3	4
Petersen 10	3	14	**20**	**150**	4	5
Kautz 12	3	24	**80**	**228**	3	7
Heawood 14	3	14	8	**378**	7	9
16-gon	3	20	48	**624**	7	13
Levi 30	3	22	76	**2520**	21	28
Hypercube 32	5	32	0	2560	16	16
Kautz 36	3	72	720	**3252**	9	31

From all optimal diameter-degree networks, Kautz networks promise the best scalable performance, even though the node count can attain only a few values. However, the performance can be fine-tuned by the number of processors per node. Inter-node CC is then implemented by message passing, whereas intra-node CC can utilize either a synchronized access to the shared L2 cache by threads or again passing messages among processes [2]. CC schedules designed by the presented evolutionary technique are targeted for micro-programmed DMA engines residing in nodes of the network. They can be easily re-programmed in case of a link failure so that CC can sustain the highest possible performance even under limited connectivity.

Some of the found CC schedules attain the theoretical lower bound on the number of communication steps and thus there is no way to improve them further. Future research may reveal limits on a size of networks that can be handled by parallel implementation of evolutionary techniques. Another direction for future research could explore a combining model for CC on Kautz networks or generalize the obtained results for Kautz networks with fat nodes. A router architecture and local (intra-node) communication could also be a subject of future optimization.

7. ACKNOWLEDGMENTS

This research has been carried out under the financial support of the research grants “Design and hardware implementation of a patent-invention machine”, GA102/07/0850 (2007-9), “Safety and security of networked embedded system applications”, GA102/08/1429 (2008-10), both care of Grant Agency of Czech Republic, and “Security-Oriented Research in Information Technology”, MSM 0021630528 (2007-13).

8. REFERENCES

[1] van der Steen, A. J., Dongarra, J. J. Overview of Recent Supercomputers. TOP 500® Supercomputer Sites, Nov. 2007 Edition, http://www.arcade-eu.org/overview/.

[2] Stewart, L. C., Gingold, D. A New Generation of Cluster Interconnect. White Paper, SiCortex Inc., Dec. 2006.

[3] Jaroš J., Ohlídal M., Dvořák V. An Evolutionary Approach to Collective Communication Scheduling, In: 2007 Genetic and Evolutionary Computational Conference, New York, US, ACM, 2007,pp. 2037-2044.

[4] Miller, M., Širáň, J. Moore graphs and beyond: A survey of the degree/diameter problem. The Electronic Journal of Combinatorics, 2005, Dynamic Survey #DS14.

[5] The (Degree, Diameter) problem for graphs. World Combinatorics Exchange, www-mat.upc.es /grup_de_grafs/grafs/taula_delta_d.html.

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

[7] Hennessy, J. L., Patterson, D.A.:Computer Architecture - A Quantitative Approach. 4th Edition, Morgan Kaufman Publishers, Inc., 2006.

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

[9] Mühlenbein, H., Paaß, G. From recombination of genes to the estimation of distributions I. Binary parameters. In Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature – PPSN IV, pp. 178-187, 1996.

[10] Jaroš, J., Dvořák, V. Speeding-up OAS and AAS Communication in Networking System on Chips, In: Proc. of 8th IEEE Workshop on Design and Diagnostic of Electronic Circuits and Systems, Sopron, HU, UWH, 2005, pp. 4, ISBN 9639364487.

[11] Ohlídal, M., Jaroš, J., Dvořák, V., Schwarz, J. Evolutionary Design of OAB and AAB Communication Schedules for Interconnection Networks, In: Lecture Notes in Computer Science, 2006, no. 3907, DE, pp. 267-278, ISSN 0302-9743.

[12] Larrañaga, P., Lozano, J. A. Estimation of Distribution Algorithms. Kluwer Academic Publishers, London 2002, ISBN 0-7923-7466-5.

[13] Goldberg D. Genetics Algorithms in Search, Optimization, and Machine Learning, Addision-Wesley Publishing Company, 1989.

[14] Dally, W., Towles, B.: Principles and Practices of Interconnection Networks, The Morgan Kaufmann Series in Computer Architecture and Design, Morgan Kaufman Publishers, 2004.