Scheduling Collective Communications on Wormhole Fat Cubes

Vaclav Dvorak
Brno University of Technology
dvorak@fit.vutbr.cz

Abstract

A recent renewed interest in hypercube interconnection network has been concentrated to the more scalable and mostly cheaper version known as a fat cube. This paper generalizes the known results on time complexity of collective communications on a hypercube for the wormhole fat cube. Examples of particular communication algorithms on the 2D-fat cube topology with 8 processors are summarized and given in detail. The performed study shows that a large variety of fat cubes can provide lower cost, better scalability and manufacturability without compromising communication performance.

1. Introduction

One of the greatest challenges faced by designers of digital systems at present is optimizing the communication and interconnection between system components. As more and more processor cores and other large reusable components have been integrated on the single silicon die (MPSoCs, Multiprocessor Systems-on-Chips, [1]), many of traditional multiprocessing techniques are modified or developed anew. The interconnection network, a fundamental component of every parallel system, and communication algorithms are no exceptions. Buses are being replaced by crossbars or by direct interconnection networks. Basically direct networks converge on the use of pipelined (wormhole) message transmission and source-based routing algorithms and the major difference among them are in topology.

The well-known binary hypercube (HC) topology is characterized by $P = 2^d$ nodes, naturally organized in d dimensions, where d is also the node degree. The worst-case distance between two nodes, (network diameter D) is logarithmic, $D = d = \log P$. The HC topology is node and edge symmetric, what simplifies the design of parallel algorithms tremendously. Computation can start in any node and the source code remains the same. Also the communication can start in any dimension. Optimal algorithms for collective communication operations exist in almost all communication models. This is why the HC topology is commonly considered the best topology there is from the algorithmic and communication point of view. The HC topology can simulate efficiently almost any other topology, too. The only drawback is its non-constant (logarithmic) degree $d=\log P$ and consequently a high number of communication channels and only partial scalability, as the number of nodes P is restricted to powers of 2.

Topologies derived from the binary HC, such as cube-connected cycles and wrap-around or ordinary butterflies [2] eliminate the drawback of non-constant node degree. They are constructed by expanding the HC vertices into cycles or linear arrays and have a small constant degree and the logarithmic diameter as before. The bisection width $2^d = P/d$ is slightly worse than the value P for hypercube and so is the scalability, since the number of processors is $P = d2^d$, i.e. only 8, 24, 64, etc.

Another useful alternative is much better scalable topology called a "fat cube" (FC). The vertices of the HC are again expanded, but now into sets of processors connected by the crossbar switch inside the router. Scalability is improved since the node can contain any number of processors, $P = m2^d$, m = 1, 2, 3, etc. The node degree grows more slowly than in the HC, $d = \log (P/m)$ and the bisection width can be adjusted by multiple links between nodes. Due to these favorable features has the FC topology been recently used e.g. in commercial DSM NUMA machine Origin 3000 (SGI). Also fat nodes with 4 Opteron processors have been used in 3D-FC connection [3] and nodes with 8 CPUs are connected into K-ring network in Swiss-T1 cluster [4]. The FC topology is also expected to appear in future networking systems for MPSoCs, because mapping FC into 2-D space is easier than in the case of the "thin" HC.

In the rest of the paper we look at the router architecture in Section 2 and present the details of

Proceedings of the 17th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD'05)
1550-6533/05 $20.00 © 2005 IEEE

hardware cost calculation in Section 3. The complexity of collective communications is analyzed in Section 4. Next Section 5 is a case study involving important collective communications on the 8-processor FC. Finally, Section 6 concludes this paper.

2. Fat cube and router architecture

Let us recall notation introduced above and establish some new notions related to the FC topology. Drawings of two instances of this topology are shown in Figure 1. We use the following parameters:

d – dimensionality of the FC/HC
D – network diameter
m – number of processors per fat node, an integer greater than 1
P – processor count $P = m2^d$ (the FC), $P = 2^d$ (the HC)
d' – dimensionality of the HC with the same number of processors as FC, $d' = \log P = d + \lceil \log m \rceil$ (binary log is the default)
f – multiplicity of external links
L – the number of external links in a FC network $L = fd2^{d-1}$. Each link consists of two channels in opposite directions.

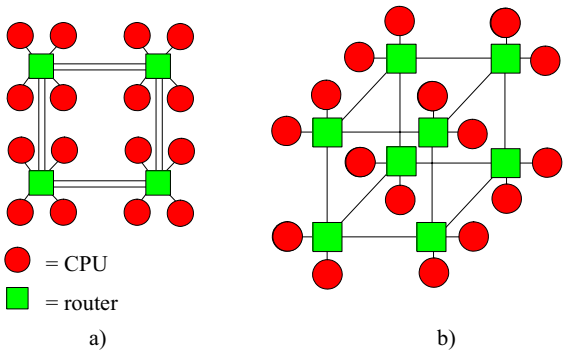


Figure 1. Examples of fat cube networks.
a) P = 16, m = 4, d = 2, f = 2
b) P = 16, m = 2, d = 3, f = 1

The design of communication algorithms depends strongly on the model used to describe the parameters of the underlying communication hardware. These models have to address key characteristics of interconnection networks, such as switching technique, channel type, message combining capability and a router model. The possible options in communication architecture are:

1. SF | WH | CS | VCT – store-and-forward, wormhole, circuit, and virtual-cut-through switching techniques
2. HD | FD | S – half-duplex, full-duplex, simplex links
3. NC | C – non-combining/ combining model capable (or not) of combining or extracting partial messages with negligible overhead
4. one-port (1) | d-port (d) – router model.

The router model for fat nodes deserves some explanation, because it is a certain generalization of the router model used in connection with thin nodes. In the simplest case, processors are connected to the router by a single link as in Fig. 1. This so called one-port model ("1") allows each of m or less processors to send a message either outside to a remote processor or to the local processor inside the same node, Fig. 2. In d-port model ("d") each processor can send up to d distinct messages simultaneously, either outside or locally. In fact, both the models are special cases of the "k-port" model, where k =1 or d. In the context of the traditional hypercube (m = 1 and f = 1) these models are known as one-port and all-port models.

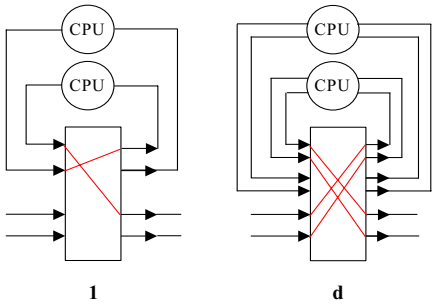


Figure 2. Router models for fat nodes.
(*m* = 2, *d* = 2, *P* = 8, *f* = 1)
1) one-port model d) d-port model

3. Cost of a fat cube network

The cost of the interconnection network has two components: the external links cost C_L and the router cost C_R. If we disregard manufacturability, the external link cost C_L can be taken simply as the number of these links $C_L = L = fd2^{d-1}$. The router cost, given mainly by the cost of $a \times b$ crossbar switch with a input ports and b output ports, is commonly taken as ab.

Let us compare the fat cube cost C and hypercube cost C' and let us find under which condition is the fat cube network cheaper. If both the networks have the same number of processors $P = P'$, then

$$P = m2^d = P' = 2^{d'} \quad or \quad d' = d + \log m .$$

The lower link cost $C_L \le C'_L$ of the fat cube

$$C_L = fd2^{d-1} \le C'_L = d'2^{d'-1} = d'm2^{d-1} \quad (1)$$

implies $fd \le md'$, what holds true because dimensionality of the fat cube with P nodes is always lower than that of a hypercube and because mostly $f \le m$.

The cost C_R of all routers together depends on the type of the port model. Table 1 compares the total router cost C_R and C_R', the product of input and output port counts, $p_{in} \times p_{out}$. Of course, we are interested especially in fat cubes with some cost advantage, i.e. when $C_R \le C_R'$. By making use of relation (1) we can transform the condition of a lower cost into inequalities involving parameters m, f and d:

1) $(m + df)^2 \le m(1 + d + \log m)^2$ (2)

d) $d^2(m + f)^2 \le 4m(d + \log m)^2$ (3)

Table 2 shows some numerically obtained solutions of inequalities (2) – (3) for $f = 1$ and 2.

For example, both 1-port fat cube networks at Fig. 1 are cheaper then 1-port hypercubes with the same number of processors P. Now the question is what will be the impact of this lower hardware cost, if any, on communication performance. We will therefore investigate the performance of collective communications on a fat cube in the next section.

Table 1. Total router cost in fat cube (C_R) and hypercube (C_R') topology

Cost	FC		HC ($m = f = 1$)	
	p_{in}, p_{out}	C_R	p_{in}, p_{out}	C_R'
1-port	$(m+df)$	$2^d(m+df)^2$	$(1+d')$	$2^{d'}(1+d')^2$
d-port	$d(m+f)$	$2^d d^2(m+f)^2$	$2d'$	$2^{d'} 4\, d'^2$

4. Complexity of collective communications on the WH fat cube

Collective communications (CCs) are frequently used in all parallel algorithms. If their overhead is excessive, performance degrades rapidly with the processor count. When we refer to „collective communications", we will assume communications involving all processors. Seven types of such collective communications are:

OAB (One-to-All Broadcast), OAS (One-to-All Scatter), AOG (All-to-One Gather), AOR (All-to-one reduce), AAB (All-to-All Broadcast), AAR (All-to-all Reduce) and AAS (All-to-All Scatter), [5]. 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). Each communication may be investigated with all possible model options, what gives too many distinct cases to explore. Therefore only the most important of them will be analyzed.

Table 2. Conditions ensuring that a fat cube be cheaper than the hypercube

$f = 1$	1	d
$m = 2$	$\forall d$	$d \le 16$
$m = 4$	$\forall d$	$d \le 8$
$m = 8$	$\forall d$	$d \le 5$

$f = 2$	1	d
$m = 2$	$d = 1$	$d \le 2$
$m = 4$	$\forall d$	$d \le 6$
$m = 8$	$\forall d$	$d \le 3$

In the rest of the paper we assume that the communication 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. Complexity of collective communication will be determined in terms of the number of these communication steps $\tau_{CC}(G)$ for the lower bound and $\tau^{CC}(G)$ for the upper bound; if network graph G is clear from the context, we will omit its symbol G (HC or FC). This figure of merit does not take into account the message length (non-uniform in combining models) or its variations from one step to another. Before analyzing communications on a fat cube, let us review the lower bounds on number of steps τ_{CC} in a hypercube network, Table 3. Lower bounds for all CCs on the WH hypercube, except OAB, are reachable by known optimal algorithms. The double-tree algorithm for OAB, Fig.3, is optimal only for $d \le 6$. Other known algorithms are nearly optimal (e.g. the algorithm by Ho-Kao, [5]).

In the following subsections we want to generalize the above results for the fat cube topology with restriction to non-combining WH models with FD links. Our approach will use the known algorithms for CC among nodes of the WH hypercube and this inter-node communication will be followed or overlapped by the local CC within the nodes on the router crossbar (intra-node communication).

Table 3. CCs on a hypercube, lower bounds τ_{CC} on time complexity

CC	1-port	all-port
OAB	$\log P\ (= d)$	$\lceil \log_{d+1} P \rceil = \lceil d/\log(d+1) \rceil$
AAB	$P - 1$	$\lceil (P-1)/d \rceil$
OAS	$P - 1$	$\lceil (P-1)/d \rceil$
AAS	$P - 1$	$P/2$

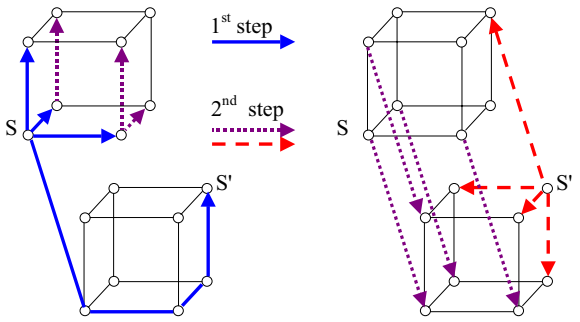


Figure 3. The "double tree" algorithm on 4D-hypercube. S, S´ are the 1st and 2nd roots.

4.1. One-to-all broadcast (OAB) on a WH fat cube

This CC is not influenced by the type of the links (HD/FD) or message (non)combining. Since just one message propagates in the network, multiple links cannot help.

Theorem 1. Complexity of OAB on the *k*-port WH fat cube measured by the number of communication steps is

$$\tau^{OAB} = \lceil \log_{k+1}(P/m) \rceil + \lceil \log_{k+1} m \rceil .$$

This upper bound can be reached for all m, $k = 1$ or d and $P/m \le 6$.

Proof.

1) k=1. OAB implemented by recursive doubling in the spanning binomial broadcast tree [5] increases the number of informed nodes twice in each of

$$D = d = \lceil \log_{k+1}(P/m) \rceil = \lceil \log 2^d \rceil$$

steps. The recursive doubling continues inside the nodes with the use of a crossbar. This intra-node communication may be overlapped with inter-node communication except the last node, so that additional $\lceil \log m \rceil$ steps are needed, q.e.d.

d) $k = d$. By making use of the double tree algorithm, that performs two partial OABs based on partial spanning binomial trees rooted in node S and S´, Fig.3, the required number of steps is $\lceil d/2 \rceil$. However,

$$\lceil d/2 \rceil = \lceil \log_{d+1} 2^d \rceil \text{ for } d \le 6.$$

The intra-node communication is done using all d ports in $\lceil \log_{d+1} m \rceil$ steps, q.e.d.

Let us note that

$$\lceil \log_{k+1}(P/m) \rceil + \lceil \log_{k+1} m \rceil \le \lceil \log_{k+1} P \rceil + 1,$$

so that the FC is in OAB never worse than the HC by more than 1 step.

4.2. All-to-all broadcast (AAB) on a WH fat cube

Optimal AAB algorithms for a hypercube matching the lower bounds in Tab.3 are based on a Hamiltonian cycle (1-port model) and on so called time-arc-disjoint spanning trees – TADTs (all-port model). All processors can use such broadcast trees synchronously with no conflicts. The following Theorem 2 establishes complexity of AAB on a fat cube, namely upper bounds τ^{AAB} in case that we do first AAB among nodes using TADTs and then AAB inside the nodes. As we will see later, due to a possible partial overlap of both the inter- and intra-node communications in d-port model, lower bound τ_{AAB} can be reached under certain conditions.

Theorem 2. Complexity of AAB on the k-port WH fat cube measured by the number of communication steps is

1) $\tau^{AAB} = \tau_{AAB}(HC) = P - 1$

k) $\tau^{AAB} = \lceil (P-m) / \min(fd, mk) \rceil + \lceil (m-1)/k \rceil P/m$

Proof.

1) We can use cyclic rotation of messages along the ring formed by the Hamiltonian cycle, m processors in every node are incorporated into that cycle. In the first step all P processors are just sending their message along the cycle and in following P-2 cycles they keep receiving and re-sending other messages. Multiple links cannot make it faster, because processors are connected to the router with a simple link.

k) Using a generic TADT rooted in every node we will perform AAB among nodes. Each node, if not a leaf, broadcasts "super-messages" consisting of m distinct messages to other nodes. In each such "super-step", m messages stored in m node processors are transferred between adjacent nodes. There are fd incoming links to a node from all dimensions and mk input links to node processors. Therefore $m(2^d - 1) = P - m$ messages destined for one node will be received in not less than $\lceil (P-m) / \min(fd, mk) \rceil$ steps. At the end will each processor have P/m distinct messages (including its own original message) to share with other local processors. As the local AAB among m nodes can be done on the router crossbar as m-1 permutations, k permutations at a time, the result is

$$\tau^{AAB} = \lceil (P-m) / \min(fd, mk) \rceil + \lceil (m-1)/k \rceil P/m,$$

q.e.d.

Provided that $fd < mk$, then $mk-fd$ ports are free during inter-node communication and can be used for broadcasting messages within the node. As there are

$(P-m)/(fd)$ steps of inter-node communication, $(P-m)(mk-fd)/(fd)$ out of total $(P/m)m(m-1)$ internal pair-wise communications can be hidden. Remaining

$$P(m-1) - (P-m)(mk-fd)/(fd)$$

pair-wise communications can be done, mk of them at a time, on mk ports. With the previous inter-node communication (and with careless handling the ceiling function) it will require

$$\tau^{AAB} = \left\lceil \frac{P-m}{fd} \right\rceil + \left\lceil \frac{P(m-1)}{mk} - \frac{P-m}{fd}\left(1-\frac{fd}{mk}\right) \right\rceil = \left\lceil \frac{P-1}{k} \right\rceil = \tau_{AAB}$$

steps. Therefore a clever overlapping of global and local communications could make an AAB algorithm as efficient as the optimal hypercube algorithm.

Contrary to OAB, combining is relevant to the complexity of AAB. There is a straightforward approach (Gather – Scatter) to combining AAB on the fat cube: one representative processor in each node gathers messages from all local peers and then AAB takes place among these representative processors with combined messages. At the end the representatives extract and distribute individual messages to local peers. We will not analyze complexity in detail, but interestingly, combining AAB can sometimes be faster on the fat cube than on the hypercube, [6].

4.3. One-to-all scatter (OAS) on a WH fat cube

This CC has similar complexity as AAB in many models. Optimal OAS algorithms for a hypercube matching the lower bounds are based on a Hamiltonian cycle (1-port model) and again on time-arc-disjoint spanning trees TADTs (d-port model). An optimal hypercube algorithm requires a broadcast tree with sub-trees of approximately equal size (±1 node). TADTs do not fulfil this requirement and must be slightly modified. The construction of such trees is known and will not be repeated here. The generic TADT tree can be rooted in any source processor and messages are sent into its sub-trees in any order. Link type (HD or FD) does not influence τ_{OAS}, rather the number of distinct messages that can be sent by the source processor in one step is important. In the fat cube topology we perform OAS among nodes first, then OAS inside nodes. Theorem 3 gives related upper bounds τ^{OAS}; for $m = f = 1$ and $k = d$ we get the lower bounds for the all-port hypercube as a special case.

Theorem 3. Complexity of OAS on the k-port WH fat cube measured by the number of communication steps is

1) $\tau^{OAS} = \tau_{OAS}(HC) = P-1$

k) $\tau^{OAS} = \lceil (P-m)/\min(fd, mk) \rceil + \lceil (m-1)/k \rceil$

Proof.

1) We can use the Hamiltonian cycle and send messages in any order to $P-1$ remote processors. We cannot use more than $f = 1$ external link, because each processor has only one internal link and both the external and internal links are connected in the Hamiltonian cycle. Therefore in $P - 1$ steps all processors will get their messages, q.e.d.

k) By making use of modified TADT for global OAS among nodes, super-messages from source node to destination nodes will consist of m messages. There are fd outgoing links from a node in all dimensions, mk output links from processors and $P - m$ messages are to be sent to other nodes. This will therefore take not less than $\lceil (P-m)/\min(fd, mk) \rceil$ steps. The local OAS in the source node requires $\lceil (m-1)/k \rceil$ steps, because the source processor can emit k messages at a time. Altogether

$$\tau^{OAS} = \lceil (P-m)/\min(fd, mk) \rceil + \lceil (m-1)/k \rceil ,$$

q.e.d. For the d-port fat cube with simple links ($f = 1$) this bound comes to $\tau^{OAS} = \lceil (P-1)/d \rceil = \tau_{OAS}$.

.

4.4. AAS on a WH fat cube

Let us recall that the optimal AAS algorithm for the 1-port hypercube matching the lower bound $\tau_{AAS} = P-1$ (see Table 3) is very simple. AAS is decomposed into $P-1$ permutations, processors with the relative address i are directly exchanging messages in step i, $i = 1, 2, \ldots, P-1$. However, the elegance of hypercube topology shows in the all-port model in which $P-1$ steps are compacted into $P/2$ steps in such a way, that all links are used in both directions in all steps! The smallest example is shown in Fig.4. Theorem 4 establishes complexity of AAS, namely upper bounds τ^{AAS} in case that we do AAS among nodes first and then inside the nodes. In some cases can these bounds be further improved by overlapping inter- and intra-node communications.

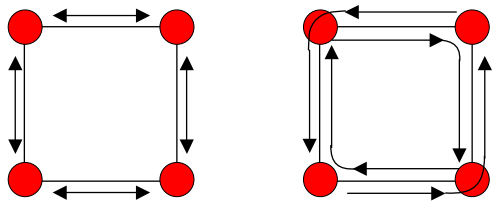


Fig. 4. AAS in 2 steps on WH 2D-HC

Theorem 4. Complexity of AAS on the k-port WH fat cube measured by the number of communication steps is

1) $\tau^{AAS} = (2^d - 1)\lceil m^2 / f \rceil$

k) $\tau^{AAS} = \lceil Pmd/[2\min(mk, fd)] \rceil + \lceil (m-1)/k \rceil$.

Proof.

1) The direct exchange HC algorithm applied to the global AAS leads to $2^d - 1$ super-steps, with exchanges of m^2 messages between each of 2^{d-1} pair of nodes, f messages at a time. One exchange super-step will thus take $\lceil m^2 / f \rceil$ steps, $f \le m$, and the whole AAS will require $(2^d - 1)\lceil m^2 / f \rceil$ steps, q.e.d.

k) We can visualize AAS as a superposition of m-to-P scatter communications by all nodes, in which each processor in the node sends P-m distinct messages outside and m-1 messages inside the node. The block of m^2 messages (a super-message) from the source node (m source CPUs in one node, each of them sending m messages to a destination node) goes through intermediate nodes to the destination node and utilizes a number of links on the way. We can count the number of channels required to connect one source node to destination nodes at all levels of the broadcast tree as

$$x = \sum_{i=0}^{d} i \binom{d}{i} = 0\binom{d}{0} + \sum_{i=1}^{d} \frac{id(d-1)!}{(d-i)!\,i\,(i-1)!} = \\ = d\sum_{i=1}^{d} \binom{d-1}{i-1} = d\sum_{j=0}^{d-1} \binom{d-1}{j} = d2^{d-1}. \qquad (4)$$

The so called communication work CW(AAS) for all 2^d nodes is thus

$$\mathrm{CW(AAS)} = 2^d x\, m^2 = d2^{2d} m^2/2 \qquad (5)$$

as each link will be used m^2-times. On the other hand, the total number of channels available at one time is a lower value of the total count of external channels $2L = fd2^d$ and the total number of output ports $2^d(mk)$, i.e. $2^d\min(fd, mk)$. Since all the external links are utilized in direct exchange algorithm in both directions in all steps, it has to hold

$$\tau^{AAS} = \lceil \mathrm{CW(AAS)}/[2^d\min(fd, mk)] \rceil = \\ \lceil Pmd/[2\min(mk, fd)] \rceil.$$

The intra-node AAS among m processors can be implemented on the router crossbar as $(m-1)$ permutations at a rate k permutations in one step, i.e. in $\lceil (m-1)/k \rceil$ steps. Together we get the desired result, q.e.d.

5. Examples of collective communication on the 8-processor, 2D-fat cube

In this section we have chosen to demonstrate communication algorithms on the small d-port fat cube with the following parameters: $d = m = 2$, $P = 8$, $f = 1$, non-combining nodes, full duplex links and wormhole switching.

5.1. One-to-all broadcast

Whereas 3 OAB steps are always needed in 8-processor hypercube using the spanning binomial tree (1+1+2+4 /1+3+3+1 processors informed in 3 steps in 1-port /d-port model), 2 steps will do in the d-port fat cube topology, see Fig. 5. The intra-node OAB is fully overlapped with 2 steps of the inter-node OAB.

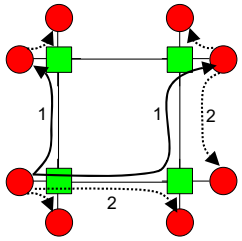


Figure 5. OAB in 2 steps on the WH fat cube

5.2. All-to-all broadcast

Theorem 3 states that we are able to complete AAB in 3+4 steps of inter- and intra-node communication, but we can do much better with their overlapping. The optimal algorithm with a full overlap of the global and local AAB is shown at Figure 6, reaching the lower bound of Theorem 2 ($f = 1$, $k = d$):

$\tau_{AAB} = \lceil (P-1)/d \rceil$ = 4 steps = max (3, 4).

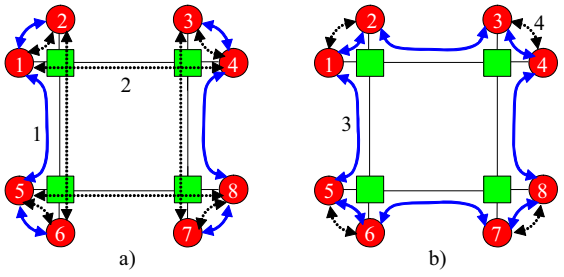


Figure 6. AAB in 4 steps on the WH fat cube
a) steps 1 and 2 b) steps 3 and 4

The path of every message from source to destination processors, divided into 4 steps, is described in Table 4.

Table 4. Four steps of the AAB communication schedule

	destination processors			
message	step 1	step 2	step 3	step 4
1	→ 2, 5	→ 6, 8	→ 7, 4	→ 3
2	→ 1, 3	→ 4, 7	→ 8, 6	→ 5
3	→ 4, 2	→ 1, 6	→ 5, 7	→ 8
4	→ 3, 8	→ 7, 5	→ 6, 1	→ 2
5	→ 6, 1	→ 2, 4	→ 3, 8	→ 7
6	→ 5, 7	→ 8, 3	→ 4, 2	→ 1
7	→ 8, 6	→ 5, 2	→ 1, 3	→ 4
8	→ 7, 4	→ 3, 1	→ 2, 5	→ 6

5.3. One-to-all scatter

In our running example (f=1, k=d) the upper bound given by Theorem 3 matches the ideal lower bound

$$\tau^{OAS} = \lceil (P-m)/d \rceil + \lceil (m-1)/d \rceil \le \lceil (P-1)/d \rceil = \tau_{OAS} = 4$$

steps, see Fig. 7. The source keeps sending messages into two sub-trees, three times 2 messages in any order and then the local OAS inside the source node is done in 1 more step.

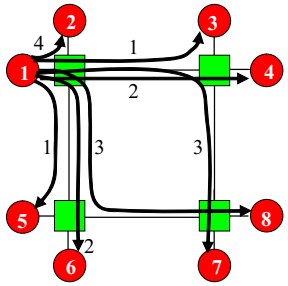


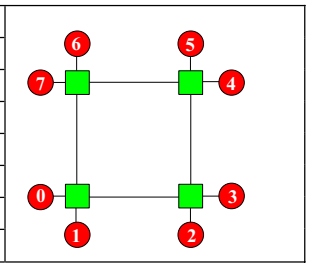
Figure 7: OAS in 4 steps on the WH fat cube

5.4. All-to-all scatter

According to Theorem 4, we should be able to complete AAS on our example fat cube in 9 steps. AAS among nodes is scheduled in 2 super-steps according to Fig.4. Considering now $m^2 = 4$ messages in a super-message, there will be 4 steps in each super-step. AAS within nodes, in our FC only exchange of messages between two processors, can be combined with any of the previous 8 steps because only one processor port is busy during inter-node communication. Pairs of processors exchanging messages in steps 1 to 8 are listed in Tab.5, local AAS communications are shown in bold.

Table 5. Eight steps of the AAS schedule

1	03, 16, 25, 47
2	02, 17, 24, 46
3	06, 12, 20, 42
4	07, 13, 21, 43
5	04, 15
6	62, 73
7	05, 14,
8	63, 72, **01, 23, 45, 67**



The performance of AAS is limited not by number of ports, but rather by the bisection width of the fat cube: AAS on the d-port FC with double links would complete in 4 steps only.

6. Results and conclusions

Summary of CC complexities for various models of our sample fat cube and hypercube networks is in Table 6. The table gives the optimized number of steps with possible overlap of global and local CCs. The communication performance of the FC is the same or better in OAB and almost the same in OAS and AAB. The AAS performance depends on multiplicity of links.

Table 6. Complexity of CCs on the 8-processor hypercube and fat cubes

m, f, d	OAB	AAB	OAS	AAS	P = 8
1, 1, 3	3	7	7	7	1-port HC
1, 1, 3	3	3	3	4	all-port HC
2, 1, 2	3	7	7	12	1-port FC
2, 1, 2	2	4	4	8	d-port FC
2, 2, 2	2	4	3	4	d-port FC

Another larger example concerns the 3D-FC with 4 CPUs per node, double links and with P = 32 processors. Table 7 gives the complexity values obtained either from Tab.3 or from Theorems 1 to 4.

Anyway, the above results concern only two particular fat cube networks, but theorems derived earlier are suitable for comparison of other configurations as well. Generally we can make the following conclusions:

1. Performance of 1-port HC and 1-port FC with the same processor count P are the same in all CCs but AAS.
2. The AAS performance is in 1-port FC proportional to $1/f$, $f \leq m$.
3. Partitioning OAB into the global and local part does not reduce the performance, but improves it by overlapping both parts.
4. Performance in OAS and AAB is poorer than in HC topology, but similar if optimization through overlapping is used or even better if multiple links are provided.
5. Poorer performance in AAS on d-port FC is given by a lower bisection width, the same performance as in the hypercube can be obtained when multiple links are used.
6. If the hardware cost is a limiting factor, then a suitable fat cube can be found which is cheaper than the equivalent hypercube with the same number of processors and with not much (if any) performance degradation.
7. The number of processors P in the fat cube configuration is not limited to powers of 2, but a power of 2 can be multiplied by an integer m. This may be more straightforward scaling than a partial hypercube.

Table 7. Complexity of CCs on the 32-processor hypercube and fat cubes

m , *f*, *d*	OAB	AAB	OAS	AAS	*P* =32
1, 1, 5	5	31	31	31	1-port HC
1, 1, 5	5	7	7	16	all-port HC
4, 1, 3	5	31	31	112	1-port FC
4, 2, 3	4	21	7	34	2-port FC
4, 1, 3	3	18	11	65	d-port FC
4, 2, 3	3	13	6	33	d-port FC
4, 4, 3	3	11	4	17	d-port FC

The future research should address other network topologies with fat nodes and links. Also other communication patterns should be studied, such as multicast and *a*-to-b broadcast or scatter. Also combining node models are of interest; partial results for SF switching have been presented in [6]. The role of combining models for WH switching should still be clarified. The research in the above directions could help optimize communication architectures for application-specific multiprocessor systems on chip, [7].

7. References

[1].Jerraya, A.A., Wolf, W., *Microprocessor Systems-on-Chips*, Elsevier Inc., 2005, ISBN 0-12385-251-X.

[2] W. Dally, B. Towles, *Principles and Practices of Interconnection Networks,* The Morgan Kaufmann Series in Computer Architecture and Design, Morgan Kaufman Publishers, 2004, ISBN: 0-12200-751-4.

[4] E. Gabrielyan, R.D. Hersch, "Efficient Liquid Schedule Search Strategies for Collective Communications", *Proc. of ICON 2004 - 12th IEEE International Conference on Networks*, Singapore, Vol. 2, November 16-19, 2004, pp 760-766.

[3] C.N. Keltcher, et al., "The AMD Opteron Processor for Multiprocessor Servers", *IEEE Micro*, March/April 2003, pp.66 – 76.

[5] J. Duato, S. Yalamanchili, L. Ni, *Interconnection Networks – An Engineering Approach*, Morgan Kaufman Publishers, 2003, ISBN 1-55860-852-4.

[6] Kutalek, V., *Performance modeling and optimization of application-specific multi-processor systems*, Ph.D. thesis, Faculty of information technology, Brno University of Technology, 2005.

[7] Dvorak,V., Communication Architectures for Application-Specific Multiprocessor Systems (on a Chip). *Proc. of the 11th International Conference on Software, Telecommunications and Computer Networks SoftCOM 2003*, Split, HR, FESB, 2003, p. 778-782.

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.