ELSEVIER Discrete Applied Mathematics 53 (1994) 79–133

DISCRETE APPLIED MATHEMATICS

Methods and problems of communication in usual networks

Pierre Fraigniaud[a,*,1], Emmanuel Lazard[b,2]

[a] *LIP-IMAG, CNRS URA 1398, Ecole Normale Supérieure de Lyon, 69364 Lyon Cedex 07, France*
[b] *LRI, CNRS URA 410, Université de Paris Sud, 91405 Orsay Cedex, France*

Received 9 August 1991; revised 29 October 1992

Abstract

This paper is a survey of existing methods of communication in usual networks. We particularly study the complete network, the ring, the torus, the grid, the hypercube, the cube connected cycles, the undirected de Bruijn graph, the star graph, the shuffle-exchange graph, and the butterfly graph. Two different models of communication time are analysed, namely the constant model and the linear model. Other constraints like full-duplex or half-duplex links, processor-bound, DMA-bound or link-bound possibilities are separately studied. For each case we give references, upper bound (algorithms) and lower bounds. We have also proposed improvements or new results when possible. Hopefully, optimal results are not always known and we present a list of open problems.

1. Introduction

Nowadays, the desire of obtaining more powerful computers leads us to use parallel architectures. One way of building such a system is to interconnect several general-purpose processors. Performances are then dependent on the balancing of computation and communication. Both are not independent: the decreasing of one often implies a raise of the other one. Two different paths have been followed by parallel computer designers: SIMD and MIMD. In the SIMD model, the machine is controlled by a central processor; this enables the designer to propose built-in communication procedures. On the contrary, MIMD computers are often built in a modular way, permitting different topologies to be used, with the drawback that, even if routers offer communication facilities, general built-in communication procedures are not available.

*Corresponding author. E-mail: pfraign@lip.ens-lyon.fr.

[1] Supported by the research program C3 and by the Direction des Recherches et Etudes Techniques.
[2] Supported by the research program C3.

0166-218X/94/$07.00
SSDI 0166-218X(93)E0043-X

Thus we are interested in describing communication algorithms dedicated to MIMD machines. If only the processors are distributed but linked to a shared memory, the main difficulty is to handle concurrent accesses to it. However, if also the memory is distributed, data movements are dependent on the topology.

In choosing a topology, the goals can be divided roughly into two categories: cost and performance. On the performance side, we look for a combination of small diameter, uniformity, extendability, short wires and redundant paths. On the cost side, we look for minimum number of wires, efficient layout, simple routing algorithms, fixed degree and a simple construction with the available technology. Notice that this wish list contains contradictions and any decision will be a compromise (see [47]). For this reason, different usual topologies have been used for designing parallel computers: hypercube, ring, torus, ...

Several articles have been published on this subject but a careful reading shows that different hypotheses are used and therefore comparisons are impossible. In this paper, we wish to give a survey of existing results, and to propose general methods of communication. We present a taxomanic framework that was previously uncodified. We also emphasize open problems which appear to be of great importance in this area. A survey has already been published by Hedetniemi et al. [45] eight years ago but, for this reason, it does not include all the recent papers on the subject. Also, our paper has a less combinatorial flavour than this previous survey as we shall use a network approach.

We invite the reader to consult the recent papers of this volume and to have a look on the special issue of *Parallel Processing Letters* on algorithmic and structural aspects of interconnection networks, to appear in the first half of 1994. Many papers related to the communication problems also appeared in the special issue of *Discrete Applied Mathematics* 37/38 (1992) on interconnection networks.

In Section 2, we present the problem, listing the different hypotheses that have been used to study it. We also give a short description of the usual networks we are interested in. As the choice of communication algorithms greatly depends on the choice of the law describing communication time, we investigate each possibility in a different section: Section 3 for the constant model and Section 4 for the linear model. Each section has a particular local organization related to the model. Finally, we conclude in Section 5.

2. Statement of the problem

For more details concerning the concepts and definitions in this section, see [91a].

2.1. Communication problems

Processors communicate by exchanging messages, but during the execution of a program, communication schemes are very dependent of the problem. Fortunately, the study of classical algorithms brings up some "generic communications" that appear very often, for example in linear or nonlinear algebra (see [24, 55]), or image processing (see [83]). In this paper, the following communication problems are analysed.

• *Broadcasting:* Sending a message from one processor to all the other ones. (One to All, ...)

• *Gossiping:* We can symmetrize the broadcast problem on all vertices: a message must be sent from each processor to all the other ones. (All to All, Total Exchange, ...)

• *Scattering:* One processor wishes to send a different message to each of the other processors. Note that the gathering problem where a processor receives a message from all the other ones is the exact inverse of scattering. Hence, only the scattering is studied. Both of these operations are useful in asymmetrical situations where one node of the network acts as a master processor and the others as its slaves. The master distributes different data sets to each of the slaves, which in turn perform computation on them. Then the master collects all results. (Personalized One to All, Distributing, ...)

• *Multiscattering:* Here again we can consider the symmetrical problem: scattering messages from each of the processors to each of the other ones. (Personalized All to All, Complete Exchange, ...)

Concerning the last three communication problems, we will assume that all involved messages are of the same length.

For other communication problems, we refer to the following papers that contain many references: [19] for *scanning*, [85] for *ranking*, [37b] and [69] for *multicasting*, [43a] and [71a] for the general *routing* problem, [12a] and [85b] for *consensus* and *synchronizer* respectively, [4a] and [39c] for *hot-potatoes* and *interval* routing respectively, [39b] for *scattering-gathering* sequences, [74aa] for perpetual gossiping, etc.

2.2. Parallel architectures

2.2.1. A network of processors

We consider a distributed memory multiprocessors system, running with a MIMD computation scheme. Memories and processors are connected by a point-to-point interconnection networks; described in terms of nodes and links.

A node in a typical network architecture consists of a processor, a memory, a fast bus, and several direct memory access (DMA) channels (see Fig. 1). Each DMA channel connects the node to one of its neighbours. The memory and DMA channels are all connected to the fast bus, and the processor is connected to the memory. A processor communicates with a neighbour by writing the information in its memory. The information is then transmitted by the appropriate DMA channel via the bus to the neighbour's memory via its bus. This communication path between two DMA-channels will be called a *link*.

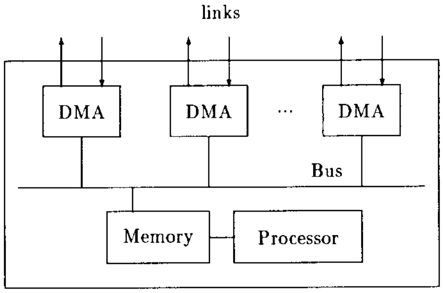

Fig. 1. Architecture of a node.

2.2.2. Communication constraints

Once we have defined the communication algorithms we are studying, we shall describe the different laws of communication we use to model real communications.

First of all, we shall consider problems where messages are sent in *store-and-forward* or *packet-switched* mode, and where a node cannot use the contents of a message until all bits are received (as opposed to *circuit-switched* mode or wormhole [26a, 28a, 37a, 85a]).

We study only the store-and-forward mode since any circuit-switched machine can also perform store-and-forward mode. Moreover, few algorithms better than the ones designed for the store-and-forward mode have been offered under the circuit-switched mode for solving intensive communication problems as broadcasting or gossiping. Seidel, Lee and Fotedar said that it is unlikely that such algorithms exist since each node must receive a copy of every message in the system [94, 95]. In fact, it strongly depends on the relationship between the several parameters of the machine as the start-up time, the bandwidth of the links or the routers delay as the recent papers [42a, 51a, 51b, 86, 93a, 96a] show.

Now, if we consider two processors p_1 and p_2 directly linked, two possibilities can arise:

(1) Only one message can travel between p_1 and p_2, from p_1 to p_2 or from p_2 to p_1. Links are then said to be *half-duplex*. It is the case, for example, when we use radio communications over one frequency.

(2) Two messages can use the link at the same time, one in each direction. Links are then said to be *full-duplex*. It is the case during phone communications.

We shall also classify communications into three types, depending on where the communication bottleneck occurs.

(1) If, during a communication, a processor can only use one of its links, we call this situation *processor-bound* because processors cannot relay quickly messages and hamper the efficiency of the network. This pattern is also called 1-*port* or *whispering*.

(2) On the contrary, when a processor can use all of its links at the same time, communications are said to be *link-bound*, because it is now the number of links which limits communications. This pattern is also called *n-ports* or *shouting*.

(3) Between these two extreme possibilities, we have the case where a processor can only use k links at the same time; communications are said to be *DMA-bound*, because what limits communications is the number of different DMA channels that can use the fast bus. Another example is when the processors have a bounded degree. With the use of crossbars and multiplexers, we can connect them via a network of greater degree, but because of the internal links, processors will not be able to use all those links at the same time.

We will use some abbreviated notation when referring to these models of communication. Specifically, $F1$ and $H1$ refer to the full-duplex and half-duplex processor-bound models respectively (the 1 indicates that a node can only communicate with one other node at a time)[1]. Full-duplex and half-duplex DMA-bound models will be denoted Fk and Hk respectively, where k indicates the maximum number of links a processor can use at the same time. Finally, $F*$ and $H*$ refer to link-bound models, where a processor can use all its links at the same time.

Now that we have listed the different possible hypotheses concerning communications, the problem is to model the communication time T to send a message from a processor to one of its neighbours. Indeed, the choice of the adapted communication algorithms will depend on this model. Many experiments show that the elementary cost T can greatly depend on the length L of the message (see [20, 23, 63, 95]). Hence, the communication time between two adjacent processors is often modeled as follows (*the linear model*):

$$T = \beta + L\tau,$$

where β is the cost of a start-up and τ is the propagation time of a data of unit length. This model allows messages to have variable length and to be split and recombined. Fraigniaud [39] has proposed a different model where parameters (start-up and propagation time) depend on the number of links used.

When we consider theoretical properties of graphs concerning communications, we shall make the assumption that the length of the messages is small so that we can simplify the linear expression of T to a constant expression. Therefore, we shall also consider the case with a time of communication between processors equal to one time unit (*the constant model*):

$$T = 1.$$

[1] Note that one can consider a *single-port* model where each processor can send (resp. receive) information to (resp. from) at most one of its immediate neighbors, but where the sending and receiving neighbors may be distinct. We do not consider this model in the following.

In this model, without changing the propagation time, messages can be split or recombined. Bertsekas et al. [17] proposed another model where the time is still constant but splitting and recombining of packets is not allowed.

Of course, there exist many other models of communication distinct from the *constant* and the *linear* models (for instance when the exchanged information should have a bounded size, when buffering is not allowed, when wormhole routin is used, ...). You will find below a non-exhaustive list of papers that present results on broadcasting, gossiping or scattering problems but in different models.

In [4a], authors suppose that processors do not know the topology of the network; closely related to that problem, in [4b, 4c, 4d] authors study the so called *radio networks*; in [17a], authors do not allow messages once dispatched ever to be delayed on their routes; in [21], authors allow only one packet to be communicated in one round; in [29, 37c, 42b, 96b], authors count 1 per packet; in [33, 102, 85d], authors do not allow buffering; in [35], authors study an intermediate model between whispering and shouting; in [50], the author considers the SIMD constraint and does not allow indirect addressing; in [64a], the author studies communications in hypergraphs (see [47a] for a survey on communication in bus networks); in [74a], authors study communication under an edge coloring model; in [74c], the author studies the broadcasting problem under the *postal* model introduced by Bar-Noy and Kipnis; in [85c], authors do not allow a message to be forwarded more than a fixed number of time; in [87a], authors study communications under *optical* facilities; in [103], authors assume a randomized length of messages; in [103a], authors consider an original approach based on matrix decomposition; in [103b], authors fix a set of costs between pairs of nodes.

2.2.3. Graph notation

The topology of the system can be described by a graph or digraph. A vertex in a graph $G = (V, E)$ corresponds to a processor together with its memory and bus, while edges or arcs represent the channels connecting the buses. More precisely, a full-duplex model corresponds to a symmetrical digraph while a half-duplex model corresponds to a graph. When G denotes an undirected graph, G^* will denote the corresponding symmetrical digraph. N will be used to denote the number of processors (vertices) in the system and $d(v)$ will denote the degree of vertex v. $\delta = \min\{d(v) \mid v \in V(G)\}$ that is the minimum degree of G and $\Delta = \max\{d(v) \mid v \in V(G)\}$ that is the maximum degree of G. If $\delta = \Delta$, the graph is said to be *regular* of degree Δ.

If u and v are two vertices of G, (u, v) will denote the arc from u to v or the edge between u and v. The *distance* between u and v in G is the length of a shortest path in G connecting u and v. The *diameter* D of G is the maximum distance between any two vertices of G.

A *vertex cut* of G is a subset V' of V such that $G - V'$ is disconnected. A *k-vertex cut* is a vertex cut of k vertices. The *connectivity* $\kappa(G)$ of G is the minimum k for which G has a k-vertex cut. G is said to be *k-connected* if $\kappa(G) \geqslant k$. An *edge cut* of G is a subset

E' of E such that $G - E'$ is disconnected. A *k-edge cut* is an edge cut of k edges. The *edge connectivity* $\lambda(G)$ of G is the minimum k for which G has a k-edge cut. G is said to be *k-edge-connected* if $\lambda(G) \geqslant k$.

2.2.4. Usual topologies

- *The complete graph* K_N: All N vertices are linked together (see Fig. 2). Thus, there is no routing problem nor choice of connection. Each vertex has degree $N - 1$. The diameter is one but the number of edges is $N(N-1)/2$, far too high to be of practical interest when N is big. However, for small values of N, complete networks have been used, for instance to interconnect i860 processors (see [84]).
- *The ring graph* C_N: It has been used for its simplicity when giving algorithms and complexity analysis. Each vertex is linked to only two neighbours, thus the degree is two (see Fig. 3). There are N edges but the diameter is $\lfloor N/2 \rfloor$, which implies a long delay when routing messages.
- *The d-torus graph* $T_N = C_{p_1} \square \cdots \square C_{p_d}$: We can generalize the previous construction to design a d-torus graph. It is a graph on $N = \prod_{i=1}^{d} p_i$ vertices, having p_i vertices in each dimension and such that each dimension can be seen as a ring (see Fig. 4). It

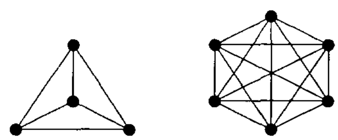

Fig. 2. Complete graphs for $N = 4$ and $N = 6$.

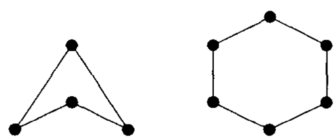

Fig. 3. Ring graphs for $N = 4$ and $N = 6$.

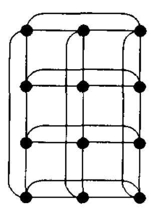

Fig. 4. A 2-torus graph with 12 vertices.

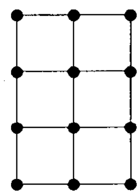

Fig. 5. A 2-grid graph with 12 vertices.

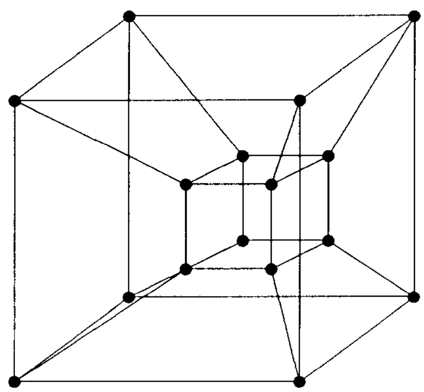

Fig. 6. A hypercube of dimension 4.

can also be seen as a cartesian product of ring graphs. The degree is therefore $2d$ and the diameter is $\sum_{i=1}^{d} \lfloor p_i/2 \rfloor$. The 2-torus has been used for instance to build the Intel Δ-prototype [88], and many transputer-based machines [77].

● *The d-grid graph* $GD_N = P_{n_1} \square \cdots \square P_{n_d}$: For $1 \leqslant i \leqslant d$, let P_{n_i} be a path on n_i vertices. The d-grid graph, also called mesh, $GD_N = P_{n_1} \square \cdots \square P_{n_d}$ is the cartesian product of those paths. Observe that any permutation of the n_i gives an isomorphic graph. This graph has degree $2d$ and diameter $\sum_{i=1}^{d} (n_i - 1)$ (see Fig. 5).

● *The hypercube graph* H_d: A hypercube H_d is a graph on $N = 2^d$ vertices, each labelled by a d-bits binary number. Edges occur between vertices whose labels differ in precisely one bit (see Fig. 6). Hypercubes may also be defined recursively as follows. A 1-dimensional hypercube is an edge with a vertex labelled 0 and the other one labelled 1. A $(d + 1)$-dimensional hypercube is constructed from two d-dimensional hypercubes, H_d^0 and H_d^1, by adding edges between each vertex in H_d^0 and the vertex in H_d^1 that has the same label and then by prefixing all of the labels in H_d^0 with a 0 and all of the labels in H_d^1 with a 1. In other words, H_d is the cartesian product of H_{d-1} and K_2.

The hypercube is used in several parallel computers such as the N-cube and the iPSC series because of its small diameter $D = d = \log_2 N$ and of its recursive definition. Furthermore, the hypercube can easily simulate other topologies and has fault-tolerance capacibilities. Unfortunately, it has $(N \log_2 N)/2$ edges, and an unbounded degree $\Delta = d = \log_2 N$.

• *The cube connected cycles graph CCC_d*: The CCC_d is a modification of the hypercube H_d obtained by replacing each vertex of the hypercube by a cycle, each cycle having d vertices. The CCC has been proposed as an interconnecting pattern for general purpose parallel processors because of its similarity with the hypercube for large class of algorithms, and because of its small bounded degree (see [87]).

The CCC_d has $d2^d$ vertices. Each one is assigned a label consisting of a pair of numbers (c, p), where c represents the cycle and satisfies $0 \leqslant c \leqslant 2^d - 1$, and p represents the position of the processor within the cycle and satisfies $0 \leqslant p \leqslant d - 1$. Let $\text{bin}(c, p)$ denote the pth bit of the binary representation of c. Then the interconnections between processors can be defined formally as follows. A processor (c, p) is connected to: $(c, (p + 1) \bmod d)$, $(c, (p - 1) \bmod d)$ and $(c + \varepsilon \times 2^p, p)$, where $\varepsilon = 1$ if $\text{bin}(c, p) = 0$ and $\varepsilon = -1$ otherwise (see Fig. 7). The degree is three.

• *The de Bruijn graph $UB(d, D)$*: The de Bruijn digraph $B(d, D)$ with indegree and outdegree d and diameter D is the digraph whose $N = d^D$ vertices are the words of length D on an alphabet of d letters. There is an arc from a vertex x to a vertex y if and only if the last $D - 1$ letters of x are the same as the first $D - 1$ letters of y, that is, there are arcs from $(x_0, \ldots, x_{D-1})$ to the vertices $(x_1, \ldots, x_{D-1}, \lambda)$ where λ is any letter of the alphabet (see Fig. 8).

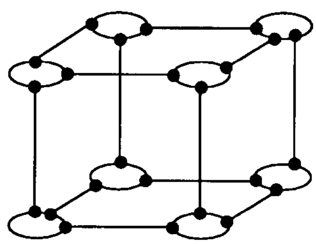

Fig. 7. The CCC_3 graph.

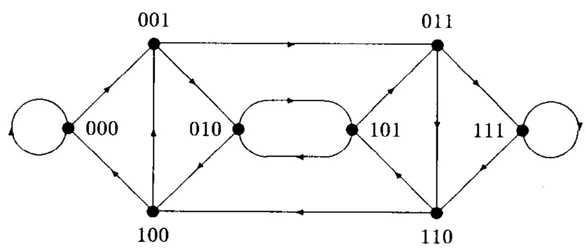

Fig. 8. The $B(2,3)$ de Bruijn graph.

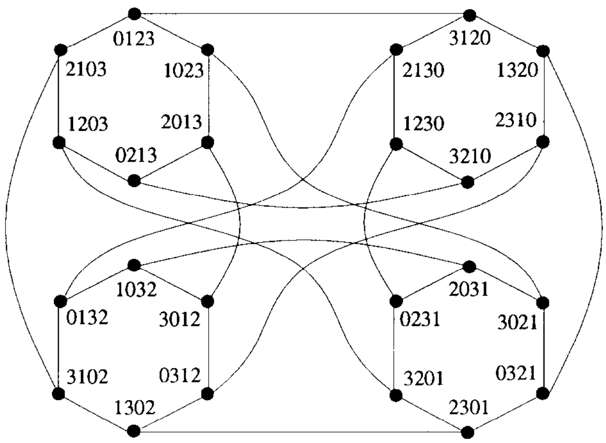

Fig. 9. The star graph S_4.

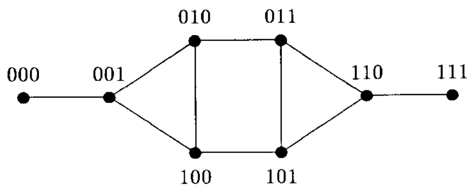

Fig. 10. The SE_3 graph.

We shall use the undirected de Bruijn graph $UB(d, D)$ obtained by removing edge orientation in $B(d, D)$. It has a degree of $2d$, a diameter of D and Nd edges. The de Bruijn graph has been proposed as good competitor for the hypercube [14].

- *The star graph* S_n: The star graph has also been proposed as an alternative to the hypercube (see [1]). S_n is the Cayley graph on the group G consisting of all permutations on n symbols, and the set of generators g defined as follows. The set g consists of $n-1$ transpositions $\{g_2, g_3, \ldots, g_n\}$ where g_i is the transposition that switches the ith element with the first and leaves the remaining elements in their same position (see Fig. 9). The star graph S_n has $N = n!$ nodes, each one has degree $n-1$. In [1], it is proved that the diameter of the star graph is $D = \lfloor 3(n-1)/2 \rfloor$.
- *The shuffle-exchange graph* SE_d: The shuffle-exchange graph is a cubic graph whose $N = 2^d$ vertices are d-bits strings $x = x_0 \ldots x_{d-1}$. Each vertex $x_0 \ldots x_{d-1}$ is connected to $x_1 \ldots x_{d-1} x_0$ and to $x_{d-1} x_0 \ldots x_{d-2}$ by *shuffle* edges and to $x_0 \ldots \overline{x_{d-1}}$ by an *exchange* edge. It is known that the shuffle-exchange graph has diameter $2d-1$ (see Fig. 10).

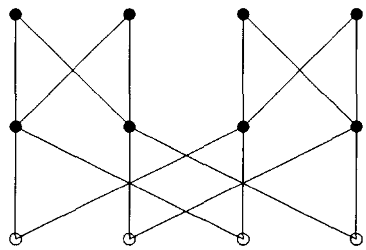

Fig. 11. The BF_2 butterfly graph.

• *The butterfly graph BF_d*: The butterfly graph BF_d is a graph whose $N = d2^d$ vertices are labelled with a pair of numbers (l, x). l is called the level $(0 \leqslant l \leqslant d-1)$ and $x = x_0 \dots x_{d-1}$ is a d-bits string called the position-within-level. Each vertex (l, x) is connected by a straight edge to $(l + 1 \bmod d, x)$ and by a cross edge to $(l + 1 \bmod d, x_0 \dots x_{l-1}\overline{x_l}x_{l+1} \dots x_{d-1})$. BF_d has degree four and diameter $\lfloor 3d/2 \rfloor$ (see Fig. 11).

2.3. Notation

Given a connected graph G and a message originator u, the broadcast time of a vertex $u, b_M(u)$, is the minimum time required to complete broadcasting from the vertex u under the M model where $H \in \{F1, H1, Fk, Hk, F*, H*\}$. The broadcast time of a graph G under model M, $b_M(G)$, is defined to be the maximum broadcast time of any vertex u in G, i.e. $b_M(G) = \max\{b_M(u) | u \in V(G)\}$. Similar definitions can be given for the gossip time $g_M(G)$, the scatter time $s_M(G)$ and the multiscatter time $m_M(G)$.

3. The constant model

"There are n ladies, and each one of them knows an item of scandal which is not known to any of the others. They communicate by telephone, and whenever two ladies make a call, they pass on to each other, as much scandals as they know at the time. How many calls are needed before all ladies know every scandal?" This problem [44], which has become known as the gossip problem, or the telephone problem, has in turn been the source of dozens of research papers that have studied problems concerning the spread of information among a set of people, whether it be by telephone calls, conference calls, letters or even computer networks.

During the period 1950–1980, parallel computers were not very common and therefore, communication algorithms were studied with a graph-theoretical approach. The first model that has been used was a constant model with a processor-bound constraint. The constant model is still frequently used, first because of its simplicity but also because it models efficiently communications when small messages are exchanged.

Sections 3.1 and 3.2 present results on the constant time broadcast and gossip problems, respectively. In both cases, we list the results under the models link-bound, processor-bound, and DMA-bound. Cases full and half duplex are distinguished in Section 3.2 (this discrimination is useless for broadcasting). As messages can be split and recombined without any loss, there is no need to distinguish between broadcasting and scattering or between gossiping and multi-scattering.

Note that we strongly invite the readers to consult the very recent paper by Hromkovič, Klasing, Monien and Peine on bounds for communication problems under the constant model [53a]. Some of the results presented in their paper might improve some bounds listed in this section.

3.1. Broadcasting

Broadcasting refers to the process of message dissemination in a communication network whereby a message, originated by one node, is transmitted to all nodes of the network. Broadcasting is accomplished by placing a series of calls over the communication lines of the network. This is to be completed as quickly as possible subject to the constraints that each call involves only one informed node and some of its neighbours, each call requires one unit of time, a vertex can participate in only one call per unit of time, and a vertex can only call its neighbours.

The first observation that can be made on broadcasting with the constant model is that speed-ups cannot be achieved by decomposing messages into packets, as the time used by a communication is independent of the length of the message sent. Therefore, during each communication between two processors, we can assume that all the information is transferred. Because of this, there is no need to distinguish between the full-duplex and the half-duplex models. That is, a broadcast scheme under a full-duplex model can be used as a half-duplex broadcast scheme without loss of execution time: as there is only one piece of information to be broadcasted, there is no need to have communications between two nodes that are already informed. Therefore communications occur only between informed and uninformed processors, thus each communication is one-way.

It is easy to see that for any vertex u in a connected graph G with N vertices, $b_{Fk}(u) \geqslant \lceil \log_{k+1} N \rceil$, since the number of informed vertices can at most be multiplied by $k+1$ during each time unit (of course, for any model M, we also have $b_M(G) \geqslant D$ in any graph of diameter D).

3.1.1. A note on mbgs

For the complete graph K_N with $N \geqslant 2$ vertices, $b_{Fk}(K_N) = \lceil \log_{k+1} N \rceil$, yet K_N is not minimal with respect to this property for any $N > 3$, while $N > k+1$. That is, we can remove edges from K_N and still have a graph G with N vertices such that $b_{Fk}(G) = \lceil \log_{k+1} N \rceil$.

The *broadcast function*, $B_k(N)$, is the minimum number of edges in any graph on N vertices such that each vertex in the graph can broadcast in minimum time, that is, in time $\lceil \log_{k+1} N \rceil$ under the model Fk. A *minimum broadcast graph* (mbg) is a graph G on N vertices having $B_k(N)$ edges and $b_{Fk}(G) = \lceil \log_{k+1} N \rceil$. Minimum broadcast graphs represent the cheapest possible communication networks (having the fewest communication links) in which broadcasting can be accomplished, from any vertex, as fast as theoretically possible.

It is not the purpose of this paper to study mbgs and the broadcast function $B_k(N)$, but historically, those were the first problems to be studied, so we shall give a short review of the results obtained in this area.

Johnson and Garey [54] showed that the problem of determining $b_{F_1}(v)$ for a vertex v in an arbitrary graph G is NP-complete (see also [82a]). Since this suggests that mbgs are extremely difficult to find, several authors have devised methods to construct graphs with small numbers of edges which allow minimum time broadcasting from each vertex. We refer to [10] for the most recent results in this domain. König and Lazard found constructions of mbgs in the Fk models for small values of N, in [60]. Liestman and Peters [74] studied $\vec{B}(N)$, the minimum number of arcs in a broadcast digraph on N vertices.

So far, the emphasis in this research has been on obtaining graphs in which each vertex can broadcast in minimum time. If these graphs are to be used in the design of actual networks, other considerations may override the need for minimum time broadcasting. In particular, some constructions result in graphs with N vertices and average degree $O(\log_2 N)$. It may be more realistic to use a graph with fixed maximum degree (see [6, 47]) in which every vertex can broadcast "quickly".

Liestman and Peters [73] first investigated broadcasting in bounded degree graphs. More recently, Bermond et al. [12], and Capocelli et al. [22] presented general lower bounds on the time required to broadcast in bounded degree graphs and reported the best known upper bounds on the time required to broadcast in bounded degree graphs. Lazard [71] generalized these results to DMA-bound bounded degree graphs. See also [36] for bounded-call broadcasting.

3.1.2. Link-bound model

This one is an easy one!

As a processor can communicate with all of its neighbours, we shall use the following algorithm: upon reception of a message, send it to all neighbours. Therefore, it is easy to see that, for any graph G of diameter D:

$$b_{F*}(G) = b_{H*}(G) = D$$

and this is of course optimal. Note however that one can be interested in eliminating redundancies; this is what is done in [95a] for the star graph.

3.1.3. Processor-bound model

Lemma 3.1. *In any graph of diameter D, if there exist three different vertices u, v_1 and v_2 with both v_1 and v_2 at a distance D of u, $b_{F1}(G) \geqslant D + 1$.*

Proof. Let G be a graph and suppose a broadcasting scheme be given for it. By induction on i, we can see that at step i of the scheme, at most one vertex at a distance i of the originator can be informed. Therefore, if we have two vertices at a distance D of the originator, only one of them can be informed in time D and we need at least one extra step to complete the broadcast: $b_{F1}(G) \geqslant D + 1$. □

Complete graph K_N: As we said in the introduction, it is easy to see that $b_{F1}(K_N) = \lceil \log_2 N \rceil$, but if we want to describe precisely the algorithm, we can use the following procedure: label each vertex from 0 to $N - 1$, 0 being the vertex initiating the broadcast. At step i, an informed vertex p sends its message to vertex $2^{i-1} + p$.

Ring graph C_N: There is only one sensible broadcast scheme: the initiator sends the message to one of its neighbours, then at each step, the two vertices which have an uninformed neighbour send the message to them. Therefore:

$$b_{F1}(C_N) = \lceil N/2 \rceil = \begin{cases} D & \text{if } N \text{ even,} \\ D + 1 & \text{otherwise.} \end{cases}$$

2-Torus graph $T_N = C_p \square C_q$: In [37], Farley and Hedetniemi investigated broadcasting in grid graphs, including 2-torus graphs. They showed that

$$\text{if } p \text{ or } q \text{ is even:} \quad b_{F1}(C_p \square C_q) = \lceil p/2 \rceil + \lceil q/2 \rceil = \begin{cases} D & \text{if } p \text{ and } q \text{ even,} \\ D + 1 & \text{otherwise,} \end{cases}$$

$$\text{if } p \text{ and } q \text{ are odd:} \quad b_{F1}(C_p \square C_q) = \lceil p/2 \rceil + \lceil q/2 \rceil - 1 = D + 1.$$

They used the following scheme: if a vertex is informed by one of its vertical neighbours, it sends the message to its other vertical neighbour. Otherwise, it first sends the message to its other horizontal neighbour, then to its upper vertical neighbour, and finally to its lower vertical neighbour.

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: Nothing has been done on broadcasting in multidimensional torus and the previous scheme cannot be easily generalized. The only known result is:

$$D \leqslant b_{F1}(C_{p_1} \square \cdots \square C_{p_d}) \leqslant D + \max(0, m - 1)$$

where m is the number of odd dimensions in $C_{p_1} \square \cdots \square C_{p_d}$. To prove this, we shall proceed by induction on d.

For $d = 2$, the result was proven in [31].

Let $d > 2$, and $C_{p_1} \square \cdots \square C_{p_d}$ be a torus, having m odd dimensions. By induction hypothesis, we can complete broadcasting in $C_{p_1} \square \cdots \square C_{p_{d-1}}$ in $D' + m' - 1$ steps, where $D' = D - \lfloor p_d/2 \rfloor$ and $m' = m$ if p_d is even and $m' = m - 1$ otherwise. Proceeding two copies of $C_{p_1} \square \cdots \square C_{p_{d-1}}$ at a time, we can complete broadcasting in p_d copies in $\lceil p_d/2 \rceil$ steps, exactly like in the ring graph. Therefore, $b_{F1}(C_{p_1} \square \cdots \square C_{p_d}) \leqslant D' + m' - 1 + \lceil p_d/2 \rceil$. If p_d is even, $m' = m$ and $\lceil p_d/2 \rceil = \lfloor p_d/2 \rfloor$. If p_d is odd, $m' = m - 1$ and $\lceil p_d/2 \rceil = \lfloor p_d/2 \rfloor + 1$. Therefore:

$$D \leqslant b_{F1}(C_{p_1} \square \cdots \square C_{p_d}) \leqslant D + \max(0, m - 1).$$

d-Grid graph $GD_N = P_{n_1} \square \cdots \square P_{n_d}$: The authors of [37] also looked at broadcasting in grid graphs. They showed that

$$b_{F1}(P_{n_1} \square \cdots \square P_{n_d}) = \sum_{i=1}^{d} n_i - d = D.$$

Hypercube graph H_d: The hypercube is a minimum broadcast graph on $N = 2^d$ vertices. To broadcast in time $\lceil \log_2 N \rceil$, use the following scheme: at step i, each informed vertex sends the message in dimension $i (1 \leqslant i \leqslant d)$. By induction on d, it can easily be shown that

$$b_{F1}(H_d) = \lceil \log_2 N \rceil = D.$$

Cube connected cycles CCC_d: It has been shown, by Meliksetian and Chen [80], that the diameter of the CCC_d is $2d + \lfloor d/2 \rfloor - 2$, for $d > 3$. A straightforward algorithm gives a broadcast time of $\lceil 5d/2 \rceil - 1$: first relay the message to the hypercube neighbour, then to the right neighbour on the ring, then to the left one.

(1) d even: then $\lceil 5d/2 \rceil - 1 = D + 1$, therefore

$$D \leqslant b_{F1}(CCC_d) \leqslant D + 1$$

with the exception of $d = 4$, where Lemma 3.1 applies, and therefore

$$b_{F1}(CCC_4) = D + 1.$$

(2) d odd: now, $\lceil 5d/2 \rceil - 1 = D + 2$, but Lemma 3.1 applies, therefore

$$D + 1 \leqslant b_{F1}(CCC_d) \leqslant D + 2.$$

de Bruijn graph $UB(d, D)$: The de Bruijn graph was first introduced as a digraph, therefore all studies were conducted on the de Bruijn digraph $B(d, D)$. Bermond and Peyrat [15] first investigated broadcasting in de Bruijn and Kautz graphs, then Heydemann et al. [46] obtained better results for broadcasting in de Bruijn digraphs or graphs, but no better bound than the ones for digraphs is known for the undirected de Bruijn graph. The results obtained are refinements of the following general statements:

Theorem 3.2 (Bermond and Peyrat). $b_{F1}(UB(2, D)) \leqslant \frac{3}{2}(D + 1)$.

Theorem 3.3 (Heydemann et al.). *$b_{F1}(UB(2,2)) = 3$, $b_{F1}(UB(4,2)) = 4$ and $b_{F1}(UB(2^p,2)) = 2p$, for $p \geqslant 3$.*

Theorem 3.4 (Heydemann et al.). *For any $D \geqslant 2$, any $d > 2$, with $2^{p-1} < d \leqslant 2^p$, $b_{F1}(UB(d,D)) \leqslant (\frac{5}{4}p + 3)D$.*

Recently, Klasing et al. [56] proved:

Theorem 3.5 (Klasing et al.). *$1.3171D \leqslant b_{F1}(UB(2,D))$.*

Star graph S_n: In [81], Mendia and Sarkar investigated broadcasting in star graphs. Their broadcast algorithm uses the recursive decomposition of the star graph. It consists of two phases, which are recursively called in each substar graph.

The first phase distributes the message from the initiator to $n-1$ other nodes in the substar, each one being connected to a different substar. Then, using the last generator g_n, the message is sent to a node in each substar. Those two phases are then repeated, in parallel, in each substar.

To distribute the message to $n-1$ other nodes in $\lceil \log_2 n \rceil$ time units, the algorithm emulates a linear ordering of the nodes and then uses the algorithm of broadcasting in the complete graph (see [43, 81]).

They showed that: $b_{F1}(S_n) \leqslant \sum_{p=2}^{n} \lceil \log_2 p \rceil + n - 2$. Very recently, Berthomé et al. [16] showed that $b_{F1}(S_n) \leqslant 2\sum_{p=2}^{\lceil n/2 \rceil} \lceil \log_2 2p \rceil + \lceil 3n/4 \rceil$ which gives:

$$\lceil \log_2 N \rceil \leqslant b_{F1}(S_n) \leqslant \lceil \log_2 N \rceil + \lceil 7n/4 \rceil + \lceil \log_2 n \rceil.$$

Shuffle-exchange graph SE_d: In [53], the authors give an algorithm to broadcast in SE_d and showed that $b_{F1}(SE_d) \leqslant 2d$. Djelloul [28] recently improved this result, and has shown that

$$b_{F1}(SE_d) = 2d - 1.$$

Butterfly graph BF_d: Stöhr was the first one to investigate broadcasting in BF_d[97, 98]. Recently, her result has been improved in [56]. The authors showed that:

$$1.7417d \leqslant b_{F1}(BF_d) \leqslant 2d - 1.$$

3.1.4. DMA-bound model

Complete graph K_N: As we said in the introduction, it is easy to see that $b_{Fk}(K_N) = \lceil \log_{k+1} N \rceil$. If we want to describe precisely the algorithm, we can use the following procedure: label each vertex from 0 to $N-1$, 0 being the vertex initiating the broadcast. At step i, an informed vertex p sends the message to vertices $(k+1)^{i-1} + pk, \ldots, (k+1)^{i-1} + (p+1)k - 1$.

Ring graph C_N: As every vertex is of degree two, we are in a link-bound situation, therefore, $b_{Fk}(C_N) = D$.

2-Torus graph $T_N = C_p \square C_q$: We complete the broadcast on one dimension, that is over one ring, which takes $\lfloor p/2 \rfloor$ time units and then we proceed with the other dimension, during $\lfloor q/2 \rfloor$ time units. Therefore, $b_{Fk}(C_p \square C_q) = D$.

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: A simple generalization of the previous case, working one dimension after the other, gives the same result: $b_{Fk}(C_{p_1} \square \cdots \square C_{p_d}) = D$.

d-Grid graph $GD_N = P_{n_1} \square \cdots \square P_{n_d}$: We have $b_{Fk}(GD_N) = b_{F1}(GD_N) = D$.

Hypercube graph H_d: We have $b_{Fk}(H_d) = b_{F1}(H_d) = D$, by using the same algorithm!

Cube connected cycles graph CCC_d: The straightforward algorithm we use in the processor-bound model now gives an upper bound of $\lfloor 5d/2 \rfloor - 1$. Therefore, we already know that $D \leqslant b_{Fk}(CCC_d) \leqslant D + 1$.

The CCC_d has degree three, so if a vertex is allowed to communicate with three or more vertices at the same time, we see that, for $k \geqslant 3$, $b_{Fk}(CCC_d) = D$.

The only remaining case is for $k = 2$. Here, the broadcast scheme is simple: once a vertex has received a message via one of its links, at the next time unit, it sends it via its two remaining links. The only problem is for the originator which has three output links. This means that one of its neighbour will be called one time unit later. The only way this can cause a problem is with the vertex (vertices) at a distance D, but a careful analysis of the scheme shows that this vertex (these vertices) can be reached by several paths passing by different neighbours of the originator and therefore, for any $k \geqslant 2$,

$$b_{Fk}(CCC_d) = D.$$

de Bruijn graph $UB(d, D)$: Of course, if $k \geqslant d$, we have $b_{Fk}(UB(d, D)) = D$. Otherwise, we shall use a theorem given in [46]:

Theorem 3.6 (Heydemann et al.). *For any de Bruijn digraph $B(d, D)$, with $d \geqslant 2$ and $D \geqslant 2$, any vertex u of the digraph and any integer $2 \leqslant p \leqslant d$, there exists a spanning p-ary tree of depth $D\lceil \log_p d \rceil$ with root at u.*

By using a k-ary tree of depth $D\lceil \log_k d \rceil$, we have the following upper bound: $b_{Fk}(UB(d, D)) \leqslant D\lceil \log_k d \rceil$. Therefore:

$$\lceil \log_{k+1} N \rceil \leqslant b_{Fk}(UB(d, D)) \leqslant \left(\frac{\log_2(k+1)}{\log_2 k}\right) \log_{k+1} N + D.$$

Star graph S_n: We saw previously how the algorithm for the complete graph could be generalized from processor-bound to DMA-bound. We use the same trick to generalize to algorithm given in [43, 81] to DMA-bound star graph. And of course, we have the same result: $b_{Fk}(S_n) \leqslant \sum_{p=2}^{n} \lceil \log_{k+1} p \rceil + n - 2$, which gives

$$\lceil \log_{k+1} N \rceil \leqslant b_{Fk}(S_n) \leqslant \lceil \log_{k+1} N \rceil + 2n - 3.$$

Shuffle-exchange graph SE_d: The SE_d has degree three, so if a vertex is allowed to communicate with three or more vertices at the same time, we see that, for $k \geqslant 3$, $b_{Fk}(SE_d) = D$.

As for the CCC, the only remaining case is for $k = 2$, thus we apply the same strategy. A careful analysis of the graph shows that the distance between two vertices is equal to the diameter only in the case of 0* and 1* (we thank J.C. König for providing an elegant proof), and those vertices have degree one. Therefore, for any $k \geqslant 2$:

$$b_{Fk}(SE_d) = D.$$

Butterfly graph BF_d: We shall use the following algorithm: during the first d steps, a vertex on level l sends the message to both neighbours on level $(l + 1) \bmod d$. After d steps, all the nodes on the same level as the originator are informed. Then, all these nodes on level l_0 send the message through straight edges to levels $(l_0 - 1) \bmod d$ and $(l_0 + 1) \bmod d$. And then each level propagates the message through successive levels. This last phase uses $\lfloor d/2 \rfloor$ steps. Therefore:

$$b_{Fk}(BF_d) = \lfloor 3d/2 \rfloor = D.$$

3.2. *Gossiping*

Before proceeding with our discussion on gossiping in usual network, we want to point out that studies similar to the ones on minimum broadcast graphs have been recently done for finding *minimum gossip graphs* (see [42, 64, 65]).

3.2.1. *Link-bound model*

3.2.1.1. Full-duplex model. Under the $F*$ model, the gossip problem on any graph has a tight bound equal to the diameter of the graph: each node can simply send its messages to all its neighbours at each step, and the time needed to gossip will simply be the diameter.

3.2.1.2. Half-duplex model.

Lemma 3.7. *If G is a bipartite graph with diameter D, then $g_{H*}(G) \leqslant D + 1$.*

Proof. Two-colour the nodes red and black and have red nodes transmit to all their neighbours on even time steps and black nodes do so on odd time steps (see [25]). $\square$

Let G be a graph of diameter D. If we choose an orientation on each edge of G, we obtain a digraph, with diameter D'. We define the oriented diameter $\vec{D}$ of G to be the minimum of all diameters D' of the digraphs obtained by all the possible edges orientations.

Lemma 3.8. *If G is a graph of diameter D and of oriented diameter $\vec{D}$, then*

$$D \leqslant g_{H*}(G) \leqslant \min(2D, \vec{D}).$$

Proof. One step in a $H*$ gossip scheme can be seen as an orientation of G, where the arcs represent directions of communication. Therefore, during two consecutive time units, we use an orientation of G and next the reverse orientation. After $2D$ time units, the gossip has been completed. Let H be an orientation of G which gives an oriented diameter $\vec{D}$. At each step, every vertex sends all its information to all of its neighbours in the digraph. After $\vec{D}$ time units, the gossip has been completed. □

Complete graph K_N: It is impossible to solve the gossip problem for the complete graph in one time unit, and a two steps solution can be easily given: first each vertex sends its information to node 0, then node 0 broadcasts all the informations.

$$g_{H*}(K_N) = 2.$$

Ring graph C_N: A ring with an even number of vertices N is bipartite, therefore Lemma 3.7 applies. Moreover, a solution in D step is impossible, therefore (see [25]),

$$\text{if } N \text{ is even,} \quad g_{H*}(C_N) = D + 1,$$

$$\text{if } N \text{ is odd,} \quad D + 1 \leqslant g_{H*}(C_N) \leqslant D + 2.$$

2-Torus graph $T_N = C_p \square C_q$: König et al. [59] studied this problem, trying to orient a torus, and applying Lemma 3.8. Table 1 summarizes their results. It shows the optimal oriented diameter of a 2-torus. Conjectures are marked with (?). They have a construction which gives the upper bound and we conjecture that the value is optimal.

Table 1

$p \mid q$	$=3$	$=4$	$=5$	$\equiv 0 \pmod 4$	$\equiv 1$	$\equiv 2$	$\equiv 3$
$=3$	$D+1$	$D+1$	$D+2$ (?)	$D+1$	$D+2$ (?)	$D+1$	$D+2$ (?)
$=4$	$D+1$	D	$D+1$	D	$D+1$	D	$D+1$
$=5$	$D+2$ (?)	$D+1$	$D+2$ (?)	$D+1$	$D+2$ (?)	D	$D+1$ (?)
$\equiv 0 \pmod 4$	$D+1$	D	$D+1$	D	D	D	D
$\equiv 1 \pmod 4$	$D+2$ (?)	$D+1$	$D+2$ (?)	D	$D+1$	D	$D+1$
$\equiv 2 \pmod 4$	$D+1$	D	D	D	D	D	D
$\equiv 3 \pmod 4$	$D+2$ (?)	$D+1$	$D+1$ (?)	D	$D+1$	D	$D+1$

This means that for nearly all 2-torus, $g_{H*}(C_p \square C_q) = D$.

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: In [59], it is proved that if two dimensions can be tightly oriented, then $\vec{D} = D$ and therefore $g_{H*}(C_{p_1} \square \cdots \square C_{p_d}) = D$.

2-Grid graph $GD_N = P_{n_1} \square P_{n_2}$: Roberts and Xu studied the problem of orienting the 2-grid in [89–91], and they showed that $D = \vec{D}$. Therefore, using Lemma 3.8:

$$g_{H*}(P_{n_1} \square P_{n_2}) = D.$$

d-Grid graph $GD_N = P_{n_1} \square \cdots \square P_{n_d}$: Gossiping under the $H*$ model in $P_{n_1} \square \cdots \square P_{n_d}$ is an *open problem*.

Hypercube graph H_d: In [61], Krumme gives algorithms for gossiping in the hypercube. He showed, using complicated strategies that, for $d \geqslant 3$, $g_{H*}(H_d) = d$. However, using Lemma 3.9, a shorter demonstration can be given for $d \geqslant 4$, by showing that an oriented hypercube can keep diameter d, see [78].

Cube connected cycles graph CCC_d: When d is even, the CCC_d is bipartite, therefore Lemma 3.7 applies. Moreover, a solution in D steps is impossible, thus, if d is even:

$$g_{H*}(CCC_d) = D + 1.$$

Gossiping under the $H*$ model in the CCC_d is an *open problem* when d is odd.

de Bruijn graph $UB(d, D)$: Of course, by using the de Bruijn digraph $B(d, D)$, and using Lemma 3.8, we see that:

$$g_{H*}(UB(d, D)) = D.$$

Star graph S_n: Recall that each node of S_n is a permutation on n elements, hence each node has a signature equal to 1 or -1. As a transposition has a signature -1, the graph is bipartite. Therefore Lemma 3.7 applies:

$$D \leqslant g_{H*}(S_n) \leqslant D + 1.$$

Shuffle-exchange graph SE_d: Djelloul [28] recently gives a gossip algorithm showing that

$$D \leqslant g_{H*}(SE_d) \leqslant \tfrac{3}{2}(D + 1).$$

Butterfly graph BF_d: When d is even, the butterfly BF_d is bipartite, therefore Lemma 3.7 applies and we have:

$$D \leqslant g_{H*}(BF_d) \leqslant D + 1.$$

Gossiping under the $H*$ model in the BF_d is an *open problem* when d is odd.

3.2.2. *Processor-bound model*

Some trivial bounds can be given for the gossip problem:

Lemma 3.9. *$\lceil \log_2 N \rceil \leqslant b_{H1}(G) \leqslant g_{H1}(G) \leqslant 2b_{H1}(G)$ and $\lceil \log_2 N \rceil \leqslant b_{F1}(G) \leqslant g_{F1}(G) \leqslant 2b_{F1}(G)$.*

Proof. The left-hand side inequalities are justified are the fact that a gossip scheme can easily be used as a broadcast scheme by sending null messages. To prove the right-hand side inequalities, use the following gossip scheme: first collect all messages in one vertex by a gather operation, then broadcast the full information to all vertices. □

3.2.2.1. Full-duplex model.

Complete graph K_N: In [57], Knödel shows that gossiping in complete graphs is nearly as easy as broadcasting. He shows that

if N is even, $g_{F1}(K_N) = \lceil \log_2 N \rceil$,

if N is odd, $g_{F1}(K_N) = \lceil \log_2 N \rceil + 1$.

Farley and Proskurowski [34] have carried out an extensive analysis of the gossip problem under the $F1$ model for rings, 2-dimensional grids, toroidal grids and Illiac grids. Their results include the following:

Ring graph C_N:

if N is even, $g_{F1}(C_N) = N/2 = D$,

if N is odd, $g_{F1}(C_N) = (N-1)/2 + 2 = D + 2$.

2-Torus graph $T_N = C_p \square C_q$:

if p and q are even,	$g_{F1}(C_p \square C_q) = D$,
if just one of p and q is odd,	$D + 1 \leqslant g_{F1}(C_p \square C_q) \leqslant D + 2$,
if both p and q are odd,	$D + 2 \leqslant g_{F1}(C_p \square C_q) \leqslant D + 4$.

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$:

$D \leqslant g_{F1}(C_{p_1} \square \cdots \square C_{p_d}) \leqslant D + 2d$.

2-Grid graph $GD_N = P_{n_1} \square P_{n_2}$:

$g_{F1}(P_3 \square P_3) = 5$,

otherwise, $g_{F1}(P_{n_1} \square P_{n_2}) = D$.

d-Grid graph $GD_N = P_{n_1} \square \cdots \square P_{n_d}$: Mahéo and Scalé [76] generalized the previous results and proved that:

if i is odd, $g_{F1}(P_3 \square \cdots \square P_3 \square P_i \square P_3 \square \cdots \square P_3) = D + 1$,

otherwise, $g_{F1}(P_{n_1} \square \cdots \square P_{n_d}) = D$.

Hypercube graph H_d: It is well known that gossiping in the hypercube is easy under the $F1$ model, using the following scheme: at step i, each vertex exchanges all its information through dimension i. By induction on d, it can easily be shown that:

$$g_{F1}(H_d) = \lceil \log_2 N \rceil = D.$$

Cube connected cycles graph CCC_d: Gossiping in ring graphs and also in the CCC_d is studied in [53]. It is proved that:

$$\text{if } d \text{ is even,} \quad D \leqslant g_{F1}(CCC_d) \leqslant 5d/2 = D + 2,$$
$$\text{if } d \text{ is odd,} \quad D + 1 \leqslant g_{F1}(CCC_d) \leqslant \lceil 5d/2 \rceil + 2 = D + 5.$$

de Bruijn graph $UB(d, D)$: Gossiping under the $F1$ model in the $UB(d, D)$ is an *open problem.*

Star graph S_n: Recently, Berthomé et al. [16] showed that $g_{F1}(S_n) \leqslant \sum_{p=1}^{n} \lceil \log_2 p \rceil + n$ which gives

$$\lceil \log_2 N \rceil \leqslant g_{F1}(S_n) \leqslant \lceil \log_2 N \rceil + 2n.$$

Shuffle-exchange graph SE_d: Gossiping under the $F1$ model in the SE_d is an *open problem.*

Butterfly graph BF_d: Djelloul [28] showed, using the fact that CCC_d is a subgraph of BF_d, that

$$\text{if } d \text{ is even,} \quad D \leqslant g_{F1}(BF_d) \leqslant \lceil 5D/3 \rceil,$$
$$\text{if } d \text{ is odd,} \quad D \leqslant g_{F1}(BF_d) \leqslant \lceil (5D + 10)/3 \rceil.$$

3.2.2.2. Half-duplex model.

Complete graph K_N: Gossiping in complete graphs with the $H1$ model was first studied by Entringer and Slater [31], who gave an upper bound on the time needed to gossip. Even and Monien [32] then showed that gossiping can be done in $p + 1$ rounds if N is even and in $p + 2$ rounds if $N \geqslant 7$ and odd, where $F(p) \geqslant \lfloor N/2 \rfloor$, $F(i)$ denoting the ith Fibonacci number. They also proved the following lower bound that differs from the upper bound by no more than one for even N and two for odd N.

$$2 + \log_\rho \lfloor N/2 \rfloor \leqslant g_{H1}(K_N)$$

where $\rho = (1 + \sqrt{5})/2$, the golden ratio. This is also stated as: $g_{H1}(K_N) \approx 1.44 \log_2 N$. These results were independently discovered by Cybenko et al. [25], Sunderah and Winkler [100], and by Labahn and Warnke [66].

Ring graph C_N: In [53], it is proved that:

$$\text{if } N \text{ is even, } N > 3, \quad g_{H1}(C_N) = N/2 + \lceil \sqrt{2N} \rceil - 1,$$
$$\text{if } N \text{ is odd,} \quad \lceil N/2 \rceil + \lceil \sqrt{2N - \tfrac{1}{2}} \rceil - 1 \leqslant g_{H1}(C_N)$$
$$\leqslant \lceil N/2 \rceil + \lceil 2\sqrt{(N+1)/2} \rceil - 1.$$

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: Cybenko et al. [25] proved the following result which is the best known for torus graphs.

Theorem 3.10 (Cybenko et al.). *Assuming that* $d \geqslant 2$, $p_1 \geqslant 8d$ *and* $p_i \geqslant 4$ *for* $i \geqslant 2$, *the gossip problem for the* $C_{p_1} \square \cdots \square C_{p_d}$ *toroidal grid under the* $H1$ *model of communication is solvable in* $D + 18d + 39$ *steps where* D *is the diameter of the grid.*

d-Grid graph $GD_N = P_{n_1} \square \cdots \square P_{n_d}$: Cybenko et al. [25] proved the following result.

Theorem 3.11 (Cybenko et al.). *Assuming that* $d \geqslant 2$ *and* $n_i \geqslant 9$ *for all* i, *the gossip problem for the* $P_{n_1} \square \cdots \square P_{n_d}$ *grid under the* $H1$ *model of communication is solvable in a number of steps equal to the diameter of the grid.*

Hypercube graph H_d: No optimal result is known for the hypercube. For a long time, only the trivial upper bound of $2d$ was known, using Lemma 3.9. Also, as for any graph, there is a lower bound of $1.44d$. Krumme [61], using complicated stategies, showed that:

$$1.44d \leqslant g_{H1}(H_d) \leqslant 1.88d.$$

Cube connected cycles CCC_d: In [53], it is proved that:

$$2d + \lfloor d/2 \rfloor - 2 \leqslant g_{H1}(CCC_d) \leqslant \lceil 7d/2 \rceil + \lceil 2\sqrt{\lceil d/2 \rceil} \rceil - 2.$$

de Bruijn graph $UB(d, D)$: Gossiping under the $H1$ model in the $UB(d, D)$ is an *open problem.*

Star graph S_n: Gossiping under the $H1$ model in the S_N is an *open problem.*

Shuffle-exchange graph SE_d: Gossiping under the $H1$ model in the SE_d is an *open problem.*

Butterfly graph BF_d: In [53], it is proved that:

$$\lfloor 3d/2 \rfloor \leqslant g_{H1}(BF_d) \leqslant \lceil 5d/2 \rceil + \lceil 2\sqrt{\lceil d/2 \rceil} \rceil - 1.$$

3.2.3. DMA-bound model

We are now using a model where a processor can use k links in each time unit: at each round, a processor can exchange messages with at most k other processors.

3.2.3.1. Full-duplex model.

Complete graph K_N: If $N \equiv 0 \pmod{k+1}$ then the vertices of K_N are numbered (i, c) where $0 \leqslant i \leqslant k$ and $1 \leqslant c \leqslant N/(k+1)$. During the first round, the $k+1$ vertices on the same "column" (that is vertices (i, c), where c is constant) exchange their messages. During round r, the $k+1$ vertices numbered $(i, c + i((k+1)^{r-1} - 1)/k)$, where $0 \leqslant i \leqslant k$, exchange their messages.

It is easily seen by induction on r that, at the end of round r, vertex $(0, c)$ knows all the information of all vertices on columns c to $(c + (k + 1)^{r-1} - 1)$. Therefore

$$g_{Fk}(K_N) = \lceil \log_{k+1} N \rceil.$$

If N is not a multiple of $k + 1$ then one can choose $(k + 1)^{\lfloor \log_{k+1} N \rfloor}$ participants and have all the others call into them on the first and last rounds, otherwise proceeding as in the above case. Therefore,

$$g_{Fk}(K_N) \leqslant \lceil \log_{k+1} N \rceil + 1.$$

Ring graph C_N: As every vertex is of degree two, we are in a link-bound situation, therefore, $g_{Fk}(C_N) = D$.

2-Torus graph $T_N = C_p \square C_q$: We complete the gossip on one dimension, that is over one ring, which takes $\lfloor p/2 \rfloor$ time units and then we proceed with the other dimension, during $\lfloor q/2 \rfloor$ time units. Therefore, $g_{Fk}(C_p \square C_q) = D$.

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: A simple generalization of the previous case, working one dimension after the other, gives the same result: $g_{Fk}(C_{p_1} \square \cdots \square C_{p_d}) = D$.

d-Grid graph $GD_N = P_{n_1} \square \cdots \square P_{n_d}$: If we consider gossiping on a path, we are in a link-bound situation, therefore, $g_{Fk}(P_N) = D$. By working one direction at a time, we prove:

$$g_{Fk}(P_{n_1} \square \cdots \square P_{n_d}) = D.$$

Hypercube graph H_d: We have: $g_{Fk}(H_d) = g_{F1}(H_d) = D$, by using the same algorithm!

Cube connected cycles graph CCC_d: By using the same algorithm as in [53], we obtain:

if d is even, $\quad 5d/2 - 2 \leqslant g_{Fk}(CCC_d) \leqslant 5d/2,$

if d is odd, $\quad 5(d - 1)/2 - 2 \leqslant g_{Fk}(CCC_d) \leqslant (5d + 1)/2.$

de Bruijn graph $UB(d, D)$: Gossiping under the Fk model in the $UB(d, D)$ is an *open problem*. However, Djelloul [28] recently showed that $g_{F2}(UB(2, D)) \leqslant 2D$.

Star graph S_n: There is no known way to perform an efficient gossiping on the star graph, so we can only use Lemma 3.8 and obtain:

$$\lceil \log_{k+1} N \rceil \leqslant g_{Fk}(S_n) \leqslant 2(\lceil \log_{k+1} N \rceil + 2n - 3).$$

Shuffle-exchange graph SE_d: Djelloul [28] showed that

$$g_{F2}(SE_d) = D.$$

Butterfly graph BF_d: If $k \geqslant 4$, then $Fk = F*$. If $2 \leqslant k \leqslant 3$, Djelloul [28] showed, using the fact that CCC_d is a subgraph of BF_d, that

if d is even, $\quad D \leqslant g_{F2}(BF_d) \leqslant \frac{5}{3}D,$

if d is odd, $\quad D \leqslant g_{F2}(BF_d) \leqslant (5D + 4)/3,$

and

$$\text{if } d \text{ is even,} \quad D \leqslant g_{F3}(BF_d) \leqslant \tfrac{5}{3}D - 2,$$

$$\text{if } d \text{ is odd,} \quad D \leqslant g_{F3}(BF_d) \leqslant \tfrac{5}{3}(D - 1).$$

3.2.3.2. Half-duplex model. Whereas for broadcasting, it was not a problem to understand the Fk definition, some different interpretations have been used for studying gossiping. At each round, a processor can exchange messages with at most k other processors. If links are half-duplex, two different rules have been used (see [32, 67, 100]):

- During each round, every processor can receive information from at most k processors and can send information to at most k processors, but it cannot both receive and send in the same round [32, 67].
- During each round, a processor can simultaneously transmit and receive (but not on the same link anyhow), provided that it can deal with at most k messages at one time [100].

Except for the complete graph, we shall always use the second hypothesis.

Complete graph K_N: These three papers studied gossiping in the complete graph and, even using these two different models, they all found the same lower bound:

$$\lceil \log_{\lambda(k)} N \rceil \leqslant g_{Hk}(K_N)$$

where $\lambda(k) = (k + \sqrt{k^2 + 4})/2$.

They also gave an upper bound which differs from this lower bound by a small additive constant.

Ring graph C_N: We are in a link-bound situation, therefore

$$\text{if } N \text{ is even,} \quad g_{Hk}(C_N) = D + 1,$$

$$\text{if } N \text{ is odd,} \quad D + 1 \leqslant g_{Hk}(C_N) \leqslant D + 2.$$

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: Gossiping under the Hk model in T_N is an *open problem.*

d-Grid graph $GD_N = P_{n_1} \square \cdots \square P_{n_d}$: Let us first consider gossiping on a path P_n. By sending all the messages to the middle node in the first $(n - 1)/2$ steps and sending them back afterwards, we see that: if n is odd, $g_{Hk}(P_n) = n - 1 = D$ and if n is even, $g_{Hk}(P_n) = n = D + 1$. Working one dimension after the other, we have:

$$D \leqslant g_{Hk}(P_{n_1} \square \cdots \square P_{n_d}) \leqslant D + m$$

where m is the number of even dimensions in $P_{n_1} \square \cdots \square P_{n_d}$.

Hypercube graph H_d: Gossiping under the Hk model in H_d is an *open problem.*

Cube connected cycles graph CCC_d: By using the same algorithm as in [53], we obtain:

if d is even, $2d + \lfloor d/2 \rfloor - 2 \leqslant g_{Hk}(CCC_d) \leqslant d/2 + 3d$,

if d is odd, $2d + \lfloor d/2 \rfloor - 2 \leqslant g_{Hk}(CCC_d) \leqslant \lfloor d/2 \rfloor + 3d + 2$.

de Bruijn graph $UB(d, D)$: Gossiping under the Hk model in the $UB(d, D)$ is an *open problem*.

Star graph S_n: Gossiping under the Hk model in S_n is an *open problem*.

Shuffle-exchange graph SE_d: If $k \geqslant 3$, then $Hk = H*$. If $k = 2$, Djelloul [28] gives a gossip algorithm showing that

if d is even, $D \leqslant g_{H2}(SE_d) \leqslant \frac{3}{2}(D + 1)$,

if d is odd, $D \leqslant g_{H2}(SE_d) \leqslant \frac{3}{2}D + \frac{1}{2}$.

Butterfly graph BF_d: If $k \geqslant 4$, then $Hk = H*$. If $k = 3$ and d even, Djelloul [28] showed, using the fact that CCC_d is a subgraph of BF_d, that

$$D \leqslant g_{H3}(BF_d) \leqslant \tfrac{5}{3}D - 1.$$

If $k = 2$, or $k = 3$ and d is odd, gossiping under the Hk model in BF_d is an *open problem*.

4. The linear model

We have seen in Section 2 that the communication cost T of sending a message from one node to one of its neighbours can greatly depend on the length of the message. It is then modeled by a sum of a *start-up* time and a *propagation* time directly proportional to the message length L, that is

$$T = \beta + L\tau.$$

The start-up β is the time to initialize the communication (acknowledgments, ...). The parameter τ is the propagation time of a unit length message (one byte for instance), $1/\tau$ is the bandwidth of the links. The fact that the communication cost depends on the message length implies that the number of steps is not the only parameter to study, the load of the links (that is the amount of information passing through them) has also to be minimized. Indeed, minimizing the number of steps is clearly efficient if the message length is small ($\beta \gg L\tau$), but for long messages minimizing the propagation time is much more efficient. Thus, in this whole section, we shall be particularly interested in describing algorithms with minimal propagation time that is algorithms reaching the minimum time complexity when the message lengths are increasing. The fastest algorithms for short messages are those described under the constant model.

Sections 4.1, 4.2 and 4.3 give algorithms dedicated respectively to broadcasting, gossiping, and both scattering and multiscattering. Contrary to the constant time model, results are organized by first listing the two cases full and half duplex. Indeed, most of the time, the link-bound algorithms are constructed from processor-bound algorithms. We do not consider the DMA-bound constraint since, as far as we know, this hypothesis has not yet been taken into account under the linear model. Moreover, we do not consider grids, shuffle-exchange and butterfly graphs since, to our knowledge, communications under the linear model have not been studied in those graphs.

4.1. Broadcasting

First, Section 4.1.1 gives general forms of lower bounds for the broadcast problem. During a broadcasting, minimizing the load of the links is obtained following two techniques: the first is pipelining and the other is finding disjoint paths between the sender and the receivers. Section 4.1.2 describes these techniques. Next, we consider the full-duplex hypothesis in Section 4.1.3, and the half-duplex hypothesis in Section 4.1.4.

4.1.1. Lower bounds

In this section, we give general forms for lower bounds on broadcasting. We mainly consider the propagation time. Expressions of the lower bounds are original. Note however that the bounds under the full-duplex model are only simple generalizations of those given in [55, 99] in the case of the hypercube. Moreover, we always use the method stated by Ho in [49]: "The minimum data transfer time can be derived considering one of the following three cases: (1) *root dominance*, that is the minimum time required for the source node to send the data, (2) *latency dominance*, that is the propagation delay for the last element, and (3) *bandwidth dominance*, that is the total bandwidth required divided by the total bandwidth available."

Proposition 4.1. *Let G be a regular graph of degree Δ and edge (or arc) connectivity λ, then*

$$b_{F1}(G) \geqslant L\tau, \qquad b_{F*}(G) \geqslant (L/\lambda)\tau,$$

$$b_{H1}(G) \geqslant \begin{cases} \dfrac{2(N-1)}{N} L\tau, & \text{if } N \text{ even}, \\ 2L\tau, & \text{if } N \text{ odd}, \end{cases} \qquad b_{H*}(G) \geqslant \frac{(N-1)}{N}\frac{2L}{\Delta}\tau.$$

Proof. The bound for the model $F1$ is obtained using the root dominance. Under the model $F*$, we use a generalization of the root dominance. Let E' be a λ-edge cut of G, and $(u, v) \in E'$. Sending a message from u to v takes at least $(L/\lambda)\tau$.

Under the half-duplex model, if the nodes can communicate simultaneously over all their links, the total bandwidth is at most $N\Delta/(2\tau)$ since the bandwidth of every edge is

$1/\tau$. If only one port can be used at any given time by each processor, the total bandwidth is at most $\lfloor N/2 \rfloor/\tau$. The total number of exchanged information on all the links during the broadcast is at least $(N-1)L$ since every vertex has to receive L data. Then the bandwidth dominance implies the proposed lower bounds. □.

Remark 4.2. Using the latency dominance, it is possible to derive other bounds for the broadcasting full duplex. Let r be a vertex of a regular graph of degree Δ, that is a source of the broadcasting and such that $\mathrm{ecc}(r) = D$. Let s be a vertex at distance D from r. During a broadcast from r, the first byte(s) reach(es) s after a time of at least $D(\beta + \tau)$. Under the $F1$ model, at most one byte has been received at this time by s and the $L-1$ others need a time of at least $(L-1)\tau$ to be received by s, while under the $F*$ model, Δ bytes could have been received at the same time by s, and the $L-\Delta$ others need a time $((L-\Delta)/\Delta)\tau$ to be received by s. Therefore:

$$b_{F1}(G) \geqslant D\beta + (D - 1 + L)\tau \quad \text{and} \quad b_{F*}(G) \geqslant D\beta + (D - 1 + L/\Delta)\tau.$$

4.1.2. *Two general techniques for broadcasting*

The following describes tools used to build fast broadcasting algorithms. In our study, we do not consider the memory control costs because the memory access and the instructions times are often negligible in front of the communication costs. However, we admit that the pipeline and the use of disjoint paths could be slightly time consuming for preparing the messages.

4.1.2.1. Pipelining. Assume that a message M of length L has to be broadcasted from node 0 of a linear array of N processors (that is an oriented ring of N processors). Under the constant model, an optimal algorithm proceeds in $N-1$ steps with a cost of $N-1$. Under the linear model, this algorithm has a cost of $(N-1)(\beta + L\tau)$. We describe a faster algorithm for large messages (see [93]). Let us divide M in packets of size B. At the first step the first packet is sent from node 0 to node 1, at the second step this first packet is forwarded from node 1 to node 2 while the second packet is sent from node 0 to node 1, and so on (see Fig. 12). The L/B packets are *pipelined* on the array from node 0 to node $N-1$. The first packet reaches node $N-1$ at time $(N-1)(\beta + B\tau)$, the following $L/B - 1$ packets successively reach node $N-1$ in a time $(L/B - 1)(\beta + B\tau)$. Thus the global cost is $(N + L/B - 2)(\beta + B\tau)$. Minimizing this cost as a function of B ($B_{\min} = \sqrt{L\beta/((N-2)\tau)}$), gives an algorithm having a cost of $(\sqrt{L\tau} + \sqrt{(N-2)\beta})^2$. Thus on a linear array of N processors, for large messages it is approximately $N-1$ times faster to use pipelining than the algorithm designed for the constant model.

Under the *link-bound model*, this technic can be easily generalized using a spanning tree of the considered network since each node can simultaneously send data to all its children in the tree. The only modification concerns the amount of start-up times which depends now on the depth of the tree (see [7, 8]):

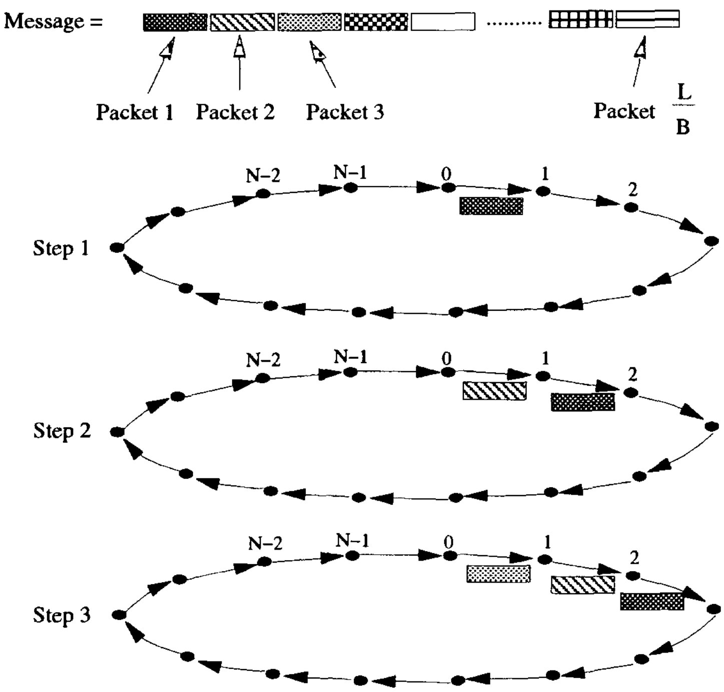

Fig. 12. Pipelining.

Proposition 4.3 [7, 8]. *Let h be the depth of a spanning tree rooted at node r of a graph or digraph G, then there exists a protocol for a link-bound broadcasting from node r whose time is* $(\sqrt{L\tau} + \sqrt{(h-1)\beta})^2$.

Now, it is also possible to construct pipelined algorithms under the *processor-bound* model. To do that, we describe the scheduling discipline in terms of labeling of the edges (or arcs) of a spanning tree of the graph (or digraph) G.

Definition 4.4. A *processor-bound labelling* of the edges (resp. arcs) of a spanning tree T of a graph (resp. digraph) G is a labeling $\text{lab}: E(T) \mapsto \mathbb{N}^*$ satisfying $\text{lab}(u, v) \neq \text{lab}(u, w)$ for any two distinct children v and w of u in the tree, and $\text{lab}(v, u) < \text{lab}(u, w)$ where v is the father of u in T and w is any child of u in T.

When we consider a unique spanning tree T of G, we can always assume without loss of generality that the processor-bound labelling of T is of the following type: the outgoing links of the root r are labelled from 1 to the degree of r in T; next if l is the label of the incoming arc of a node u in the tree, u having c children, then its outgoing arcs are labelled from $l + 1$ to $l + c$.

The problem consists then to minimize the maximum label, indeed it must be as small as possible to insure a fast broadcast.

Proposition 4.5. *Let G be a graph or digraph and T be a spanning tree of G rooted in r. Let Δ_T be the maximum degree of T. Assume we know a processor-bound labelling of T of maximum label ω. Then there exists a protocol for a processor-bound broadcasting from node r using the tree T whose time is* $(\sqrt{\Delta_T L\tau} + \sqrt{(\omega - \Delta_T)\beta})^2$.

Proof. We show how to pipeline through the tree T. The packets are sent every Δ_T steps. Thus an edge labelled l is used at steps $l, l + \Delta_T, l + 2\Delta_T, \ldots$. This avoids collisions. Now, the cost of such a pipelined algorithm (with packets of size B) is $\omega(\beta + B\tau)$ for the first packet to reach the extremity of the arc labelled ω, plus $\Delta_T(L/B - 1)(\beta + B\tau)$ for the $L/B - 1$ other packets following the first one to reach the extremity of the arc labelled ω. Choosing $B_{\min}$ minimizing the global time, the pipelined algorithm has a cost of $(\sqrt{\Delta_T L\tau} + \sqrt{(\omega - \Delta_T)\beta})^2$. □

Note that since ω is the maximum label, a non-pipelined algorithm performs in ω steps with a cost of $\omega(\beta + L\tau)$. Thus, for large messages, the pipelined algorithm is about ω/Δ_T faster than the non-pipelined one.

However, in both cases (link-bound and processor-bound) the pipeline technic is not sufficient to construct optimal broadcasting algorithms (compare with the bounds of Proposition 4.1).

4.1.2.2. Disjoint spanning trees. If the messages are sufficiently long, it is efficient to use disjoint paths between the source and the receivers. For instance finding p spanning trees rooted at a same node r, and pairwise arc-disjoint (resp. edge-disjoint) allows to define fast broadcasting algorithm under the full-duplex (resp. half-duplex) model of communication: the message is cut in p blocks of length L/p, and each block is broadcasted through a different tree. In the following, we will denote ADST for "arc-disjoint spanning tree" and EDST for "edge-disjoint spanning tree".

Two theorems of graph theory give sufficient condition for the existence of disjoint spanning trees in any graph. The first one, due to Kundu [62], applies to the half-duplex model of communication. The second one, due to Edmonds [30], applies to the full-duplex model of communication.

Theorem 4.6 (Kundu) [62]. *Let G be a graph with edge connectivity* $\lambda = 2p$, *then there exist at least p edge-disjoint spanning trees in G.*

Theorem 4.7 (Edmonds) [30]. *Let G be a directed graph with arc connectivity λ, and let u be any node in G. There exist at least λ arc-disjoint spanning trees in G all rooted at node u.*

Under the link-bound model, giving p ADST's or EDST's rooted at r is enough to define a broadcasting algorithm from r, using the pipelining technic independently on each tree (see [7, 8]):

Proposition 4.8 [7, 8]. *Let $h(p)$ be the maximum depth of p arc-disjoint spanning tree (resp. edge-disjoint spanning trees) rooted at node r of a digraph (resp. graph) G, then there exists a protocol for a link-bound broadcasting from node r whose time is at most* $(\sqrt{L\tau/p} + \sqrt{(h(p) - 1)\beta})^2$.

Thus under the link-bound model, the aim is to find as many disjoint trees as possible. In particular, for a digraph of connectivity λ, we know that it is possible to find λ ADST's, and thus there exists a protocol for a link-bound broadcasting reaching the minimum propagation time for large messages (compared to Proposition 4.1).

However, we are also interested in minimizing the start-up times, which are related to the maximum depth of the trees. There exist many algorithms to find as many ADST's as possible, but they do not give bounds on the depth of the trees obtained (see [40a]). In fact, finding disjoint spanning trees of minimum depth, rooted at a given vertex has been proved to be NP-complete by Alon (see [9] for his proof).

Under the processor-bound model, there are many problems of contention. Indeed the scheduling of all the trees must be simultaneously considered since we have to insure that only one communication link is used by each processor at a given time. This problem is even much more complicated when we want to pipeline the messages through all the trees. However, we can still describe a scheduling in terms of labelling:

Definition 4.9. A *processor-bound disjoint-labelling* of the edges (resp. arcs) of p EDST's (resp. ADST's) of a graph (resp. digraph) G is a processor-bound labelling lab_i of each tree T_i, $i = 0, \dots, p-1$ such that for every node u, $\mathrm{lab}_i(u, v) \neq \mathrm{lab}_j(u, w)$, $\mathrm{lab}_i(v, u) \neq \mathrm{lab}_j(w, u)$, and $\mathrm{lab}_i(u, v) \neq \mathrm{lab}_j(w, u)$, $\forall v \neq w$, and $\forall i, j = 0, \dots, p-1$. We define the *delay* of a processor-bound disjoint-labelling of p EDST's (resp. p ADST's) as the smallest integer ϱ such that, for every node u, for any couple of edges (resp. of non-symmetrical arcs) of extremity u, these edges (resp. arcs) have a distinct label modulo ϱ.

Note that it is possible for two symmetric arcs of a symmetric digraph to have a same label modulo the delay. Indeed, the processor-bound model is described in terms of links, eventually corresponding to two arcs under the full-duplex model. Following the above definition, we get:

Proposition 4.10. *Let G be a graph (resp. digraph) having p EDST's (resp. ADST's), all rooted in r. Assume we know a processor-bound disjoint-labelling of the p trees of* *maximum label* $\omega(p)$, *and of delay* ϱ. *There exists a protocol for a processor-bound broadcasting from node r whose time is* $(\sqrt{\varrho L\tau/p} + \sqrt{(\omega(p) - \varrho)\beta})^2$.

Proof. The message is cut in p blocks of size L/p and each block is pipelined through each tree. To insure the use of only one link by each node at any time, the packets are sent every ϱ steps. The cost of the algorithm (with packets of size B) is $\omega(p)(\beta + B\tau)$ for the first packet to reach the extremity of the arc labelled $\omega(p)$, plus $\varrho(L/(pB) - 1)(\beta + B\tau)$ for the $L/(pB) - 1$ other packets following the first one to reach the extremity of the arc labelled $\omega(p)$. Choosing $B_{\min}$ minimizing the global time, the pipelined algorithm has a cost of $(\sqrt{\varrho L\tau/p} + \sqrt{(\omega(p) - \varrho)\beta})^2$. □

The aim is then to minimize the delay. More precisely, if it is possible to construct a processor-bound disjoint-labelling of p disjoint trees with a delay $\varrho = p$, then there exists a protocol for a processor-bound broadcasting of large messages reaching the minimum propagation time, that is performing in $(\sqrt{L\tau} + \sqrt{(\omega(p) - p)\beta})^2$. Note that, it could be useless to find a maximum number of disjoint spanning trees. For instance, for any hamiltonian graph G of order N even, there exists a broadcasting protocol in G^* using the two spanning trees consisting of the two directed hamiltonian paths, and reaching the minimum propagation time for large messages: we label the hamiltonian circuits from the source from 1 to $N - 1$ in one direction, and from 2 to N in the other direction (see Fig. 13). Since N is even, the delay is 2. Thus following Proposition 4.10, we can broadcast in $(\sqrt{L\tau} + \sqrt{(N - 2)\beta})^2$. Note however that such a protocol is interesting only for very large messages due to the important amount of start-up times. To decrease the global start-up cost, we still have to find disjoint spanning trees of small depth.

In the following, we shall either give explicitly a processor-bound disjoint-labelling of p ADST's or EDST's of delay ϱ and maximal label $\omega(p)$, or prove that there exists such a labelling and give only upper bounds on ϱ and $\omega(p)$. To find upper bounds, we

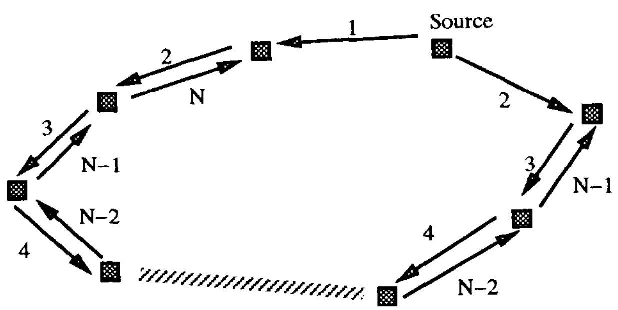

Fig. 13. Processor-bound disjoint-labelling of delay 2: two hamiltonian paths in a symmetrical digraph of even order.

shall use the *chromatic index* [5] of the studied graph G, that is the minimum number q such that it is possible to label all the edges of $E(G)$ with numbers in $\{0, \ldots, q-1\}$, each node having all its links labelled with different numbers:

Corollary 4.11. *Let $q(G)$ be the chromatic index of G, and $h(p)$ be the maximal depth of p arc-disjoint spanning trees or edge-disjoint spanning trees of G rooted at some node r. There exists a protocol for a processor-bound broadcasting from node r with time less than $(\sqrt{q(G)L\tau/p} + \sqrt{h(p)(q(G)-1)\beta})^2$.*

Proof. If G has chromatic index $q(G)$, then there exists a processor-bound disjoint-labelling of the p trees of delay ϱ, $1 \leqslant \varrho \leqslant q(G)$, and maximum labelling $\omega(p) \leqslant q(G) + (h(p)-1)(q(G)-1) = h(p)(q(G)-1)+1$. The broadcasting time then follows using Proposition 4.10. □

Finally, instead of using arc-disjoint trees, we could use time-disjoint broadcast trees (meaning that at a given time, one arc is used in at most one broadcast tree). However these trees are more difficult to find and do not allow pipelining.

In the following, we shall describe, for each topology, as many edge- or arc-disjoint spanning trees as possible for the considered graph or digraph. The trees are defined using functions children(u, i) and/or father(u, i) which give the children and/or the father of node u in a tree T_i, $i = 0, 1, 2, \ldots$

4.1.3. Full-duplex model

Complete graph K_N: Under the link-bound model $F*$, the algorithm defined in Section 3 has a cost of $\beta + L\tau$. Assume without loss of generality, the source to be node 0. Let us define $N-1$ spanning trees of K_N^* labelled from 1 to $N-1$ (see Fig. 14):

$$\text{children}(u, i) = \begin{cases} i & \text{if } u = 0; \\ v, \forall v \neq i \text{ and } v \neq 0 & \text{if } u = i; \\ \emptyset & \text{if } u \neq i \text{ and } u \neq 0. \end{cases}$$

It is straightforward that these $N-1$ trees are arc-disjoint and of maximum depth 2 (note that this depth is optimal for $N-1$ arc-disjoint trees of K_N^*). Thus following Proposition 4.8:

$$b_{F*}(K_N) \leqslant (\sqrt{L\tau/(N-1)} + \sqrt{\beta})^2.$$

Under the processor-bound model $F1$, the algorithm defined in Section 3 has a cost of $\lceil \log_2 N \rceil(\beta + L\tau)$. We consider the $N-1$ arc-disjoint spanning trees described before, and we give a processor-bound disjoint-labelling $\text{lab}_i(u, v)$, $\forall i \in \{1, \ldots, N-1\}$ by $\text{lab}_i(u, v) = u + v$, $\forall v \in \text{children}(u, i)$. It is easy to see that the maximum label is $\omega(N-1) = 2N-3$, and the delay is $\varrho = N-1$. Thus, following Proposition 4.20, we get:

$$b_{F1}(K_N) \leqslant (\sqrt{L\tau} + \sqrt{(N-2)\beta})^2.$$

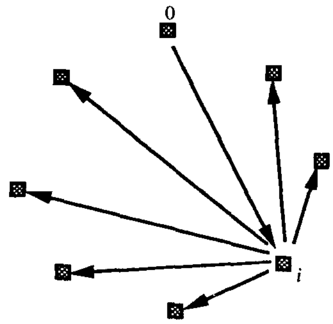

Fig. 14. The ith tree of the $N-1$ spanning trees of K_N^*.

This algorithm implies an important amount of start-up times. In fact, it may be faster to use hamiltonian cycles (see Section 4.1.2).

Ring graph C_N: Under the link-bound model $F*$, a simple algorithm runs in $\lfloor N/2 \rfloor$ steps in a time $\lfloor N/2 \rfloor(\beta + L\tau)$. There are clearly two arc-disjoint spanning trees of depth $N-1$ rooted at a given vertex r of C_N^* (a clockwise tree and a counterclockwise tree [38]). Following Proposition 4.8, they allow to broadcast in a time:

$$b_{F*}(C_N) \leqslant (\sqrt{L\tau/2} + \sqrt{(N-2)\beta})^2.$$

Under the processor-bound model $F1$, a simple algorithm runs in $\lceil N/2 \rceil$ steps in a time $\lceil N/2 \rceil(\beta + L\tau)$ (see Section 3). We have to define a processor-bound labelling of the clockwise and the counterclockwise spanning trees. If N is even, we refer to Fig. 13, otherwise a processor-bound disjoint-labelling of two ADST's of C_N^* of delay 2 does not exist. Thus, following Proposition 4.10 when N is even and Corollary 4.11 when N is odd, we get:

$$b_{F1}(C_N) \leqslant \begin{cases} (\sqrt{L\tau} + \sqrt{(N-2)\beta})^2 & \text{if } N \text{ even,} \\ (\sqrt{\tfrac{3}{2}L\tau} + \sqrt{2(N-1)\beta})^2 & \text{if } N \text{ odd.} \end{cases}$$

Thus, we reach the minimum propagation time only if N even. The exact complexity of the broadcast in C_N under the full-duplex processor-bound model for odd orders is not known.

2-Torus graph $T_N = C_p \square C_q$: Fraigniaud [38] has described four ADST's in $C_p^* \square C_q^*$ maximal depth $p+q-2$. Assume the rows to be labelled from 0 to $p-1$, and the columns from 0 to $q-1$. Assume, without loss of generality, the source of the broadcasting to be $(0,0)$. The father function of the tree T_{East}

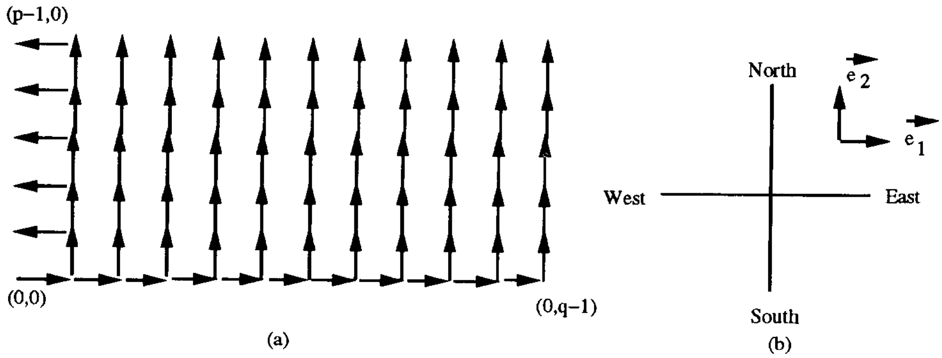

Fig. 15. The tree T_{East} of $C_p^* \square C_q^*$.

is (see Fig. 15 (a)):

$$\text{father}((l,c), T_{\text{East}}) = \begin{cases} \emptyset & \text{if } l = c = 0; \\ (0, c-1) & \text{if } l = 0 \text{ and } c \neq 0; \\ (l-1, c) & \text{if } l \neq 0 \text{ and } c \neq 0; \\ (l, 1) & \text{if } l \neq 0 \text{ and } c = 0. \end{cases}$$

The trees T_{North}, T_{West}, and T_{South} are obtained by rotation. It is easy to check that these four trees are arc-disjoint, and of maximal depth $p + q - 2$. It is possible to do better: Michallon et al. [82] have constructed four ADST's of $C_p^* \square C_q^*$ of maximal depth $D + 1 = \lfloor p/2 \rfloor + \lfloor q/2 \rfloor + 1$. Proposition 4.8 then implies:

$$b_{F*}(C_p \square C_q) \leqslant (\sqrt{L\tau/4} + \sqrt{D\beta})^2.$$

To give an upper bound of the processor-bound broadcasting time, we use the chromatic index of $C_p \square C_q$:

Lemma 4.12. *The chromatic index of $C_p \square C_q$ is 4 if p or q is even, and 5 if p and q are odd.*

Proof. It is easy to label the edges of $C_p \square C_q$ with four numbers if p or q is even. If p and q are odd, the total number of vertices is odd, thus the chromatic index cannot be 4. It is easy to label the edges with 5 numbers. □

Thus, following Corollary 4.11 and the results in [82], we get:

$$b_{F1}(C_p \square C_q) \leqslant \begin{cases} (\sqrt{L\tau} + \sqrt{3(D+1)\beta})^2 & \text{if } p \text{ or } q \text{ is even,} \\ (\sqrt{\tfrac{5}{4}L\tau} + \sqrt{4(D+1)\beta})^2 & \text{if } p \text{ and } q \text{ are odd.} \end{cases}$$

The exact complexity of the broadcast in $C_p \square C_q$ under the full-duplex processor-bound model for odd p and q is not known.

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: We can describe the four spanning trees of $C^*_{p_1} \square C^*_{p_2}$, T_{East}, T_{North}, T_{West}, and T_{South} in terms of directions: let $e_1 = (1, 0)$ and $e_2 = (0, 1)$ (see Fig. 15(b)), then T_{East} can be described by the pair (e_1, e_2):

$$T_{\text{East}} = (e_1, e_2).$$

That is (see Fig. 15):

(1) From $(0, 0)$, cover all the arcs in the e_1 direction.

(2) From $\{(e_1)\}$, cover all the arcs in the e_2 direction except the ones of $\{(e_2)\}$, where $\{(e_i)\}$, $i = 1, 2$ denotes the space generated by e_i.

(3) Complete the tree in the $-e_1$ direction to reach $\{(e_2)\}$.

Using R, the $\pi/2$-rotation, the other trees are obtained by applying this operator respectively 1, 2, and 3 times on T_{East}:

$$T_{\text{North}} = (e_2, -e_1);$$

$$T_{\text{West}} = (-e_1, -e_2);$$

$$T_{\text{South}} = (-e_2, e_1).$$

We can generalize this construction, a tree $T_1 = (e_1, \ldots, e_d)$ is constructed in $d + 1$ steps:

Step 1: from $(0,0)$ cover all the arcs in the e_1 direction.

Steps i, $2 \leqslant i \leqslant d$: from $\{(e_1, \ldots, e_{i-1})\}$ cover all the arcs in the e_i direction except the ones of $\{(e_2, \ldots, e_i)\}$.

Step $d + 1$: complete the tree in the $-e_1$ direction to reach $\{(e_2, \ldots, e_d)\}$.

The $2d - 1$ other trees are obtained by the linear operator

$$R = \begin{pmatrix} 0 & & & & & -1 \\ 1 & 0 & & & & 0 \\ 0 & 1 & . & & & 0 \\ & 0 & . & . & & . \\ & & . & . & 0 & 0 \\ & & & 0 & 1 & 0 \end{pmatrix}$$

satisfying $R^{2d} = \text{Id}$. It is easy to check by induction that the trees are arc-disjoint (the trees use distinct directions at each step, and there is no collision with directions used at previous step). The depth of these trees is $\sum_{i=1}^{d} p_i - d \leqslant 2D$, therefore Proposition 4.8 implies:

$$b_{F*}(C_{p_1} \square \cdots \square C_{p_d}) \leqslant (\sqrt{L\tau/(2d)} + \sqrt{(2D-1)\beta})^2.$$

Lemma 4.13. *The chromatic index of $C_{p_1} \square \cdots \square C_{p_d}$ is $2d$ if one of the cycles has even length, and $2d + 1$ if all the cycles are of odd order.*

Corollary 4.11 then applies:

$$b_{F1}(C_{p_1}\square\cdots\square C_{p_d}) \leqslant \begin{cases} (\sqrt{L\tau}+\sqrt{2D(2d-1)\beta})^2 & \text{if one of the } p_i\text{'s is even,}\\ \left(\sqrt{\dfrac{2d+1}{2d}L\tau}+\sqrt{4dD\beta}\right)^2 & \text{if all the } p_i\text{'s are odd.}\end{cases}$$

The trees built above have the advantage to be simple to design locally. Note that it has been recently shown that it is possible to construct $2d$ arc-disjoint spanning trees in $C^*_{p_1}\square\cdots\square C^*_{p_d}$ of maximum depth $D+\max_i\lceil p_i/2\rceil+1$ (see [40a, 68]). This allows to improve the start-up times, but these trees are more tricky.

The exact complexity of the broadcast in $C_{p_1}\square\cdots\square C_{p_d}$ under the full-duplex processor-bound model when all the p_i's are odd is not known.

Hypercube graph H_d: The main results for this topology under the link-bound model are given in [55, 99]. We mainly refer to [55] where Ho and Johnsson define d ADST's rooted at any given vertex of H^*_d. The maximum depth of these trees is $d+1$ (and is therefore optimal). Thus they obtain a broadcasting time of $(\sqrt{L\tau/d}+\sqrt{d\beta})^2$. However, Stout and Wagar propose in [99] another approach which allows to decrease the start-up time. Their algorithm emulates the disjoint spanning trees by a modification of the last sendings (see [39]):

$$b_{F*}(H_d) \leqslant (\sqrt{L\tau/d}+\sqrt{(d-1)\beta})^2.$$

In [55], Ho and Johnsson give a processor-bound labelling of their d ADST's of H_d. The maximum label is $\omega(d)=2d$ and the delay is $\varrho=d$. Thus:

$$b_{F1}(H_d) \leqslant (\sqrt{L\tau}+\sqrt{d\beta})^2.$$

Although designed for hypercubes with store-and-forward message passing, this algorithm has been implemented on the iPSC/2 and iPSC/860 which provide circuit-switched message passing (see [52, 95]): obtained performances were quite good! Moreover, this algorithm can also be used to broadcast in a SIMD machine [50].

Finally, note that similar constructions have been described for topologies similar to the hypercube. For example, in the folded hypercube [70] F_d, which is a hypercube H_d enhanced with extra links between antipodal nodes, Ho constructs $d+1$ ADST's of F^*_d (see [48]). Concerning the k-ary d-cube [26], MacKenzie and Seidel give a construction of $(k-1)d$ ADST's of maximal depth $d+1$ (see [75]). Tien et al. [101] have studied the broadcast problem on incomplete hypercubes, etc.

Cube connected cycles graph CCC_d: Fraigniaud and Ho [40] have constructed three ADST's rooted at any given vertex of CCC^*_d of maximal depth $4d$. Thus following Proposition 4.8:

$$b_{F*}(CCC_d) \leqslant (\sqrt{L\tau/3}+\sqrt{(4d-1)\beta})^2.$$

Lemma 4.14 (see Fig. 16). *The chromatic index of CCC_d is* 3.

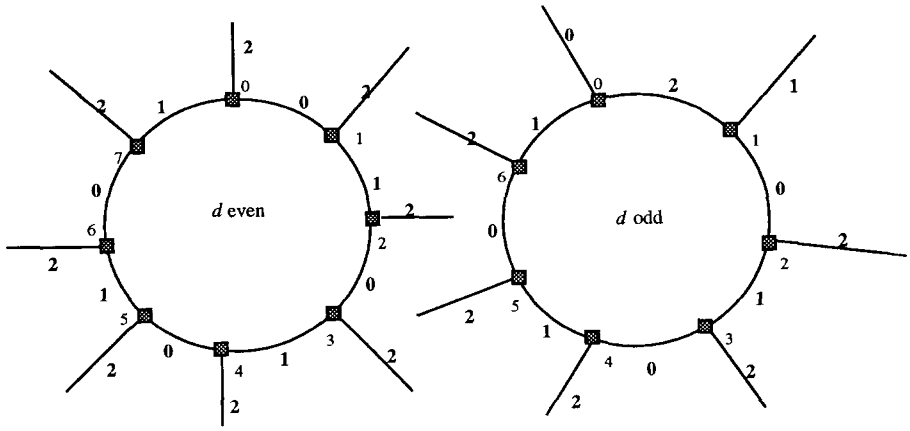

Fig. 16. The chromatic index of CCC_d is 3.

This lemma and Corollary 4.11 imply:

$$b_{F1}(CCC_d) \leqslant (\sqrt{L\tau} + \sqrt{8d\beta})^2.$$

de Bruijn graph $UB(d, D)$: Bermond and Fraigniaud [7] have constructed $2d - 2$ ADST's rooted at any given vertex of $UB^*(d, D)$ of maximal depth $D + 2\lfloor D/2 \rfloor + 1$. Thus, following Proposition 4.8:

$$b_{F*}(UB(d, D)) \leqslant (\sqrt{L\tau/(2d-2)} + \sqrt{2D\beta})^2.$$

Lemma 4.15 (Bermond and Hell [11]). *The chromatic index of* $UB(d, D)$ *is* $2d$.

Thus, applying Corollary 4.11:

$$b_{F1}(UB(d, D)) \leqslant (\sqrt{(d/(d-1))L\tau} + \sqrt{2D(2d-1)\beta})^2,$$

which is close to the lower bound when the degree increases. For small d, the exact complexity of the broadcast in $UB(d, D)$ under the full-duplex processor-bound model is not known.

Star graph S_n: For broadcasting in S_n^*, MacKenzie and Seidel [75] have constructed $n - 1$ ADST's rooted at any given vertex of S_n^* of maximal depth $3n - 4$. Thus:

$$b_{F*}(S_n) \leqslant (\sqrt{L\tau/(n-1)} + \sqrt{(3n-5)\beta})^2.$$

Clearly, the chromatic index of S_n is $n - 1$, thus Corollary 4.11 applies:

$$b_{F1}(S_n) \leqslant (\sqrt{L\tau} + \sqrt{(n-2)(3n-4)\beta})^2.$$

We refer to [43b] for a comparison between star graph and hypercubes relative to the broadcasting problem.

4.1.4. Half-duplex model

Since most of the existing multiprocessors support full-duplex communications, broadcasting under the half-duplex model has been less studied. However, some synchronization problems can decrease the performances under the full-duplex model (see [95]), so we think that it is of the main interest to describe algorithms designed for the half-duplex model.

Complete graph K_N: Assume without loss of generality the source to be node 0. Let us define $\lfloor N/2 \rfloor$ spanning trees of K_N labelled from 1 to $\lfloor N/2 \rfloor$ (all sums are modulo N):

$$\text{children}(u, i) =$$

$$\begin{cases} i & \text{if } u = 0, \\ v, \forall v = i+1, \ldots, i + \lceil N/2 \rceil - 1 & \text{if } u = i, \\ v, \forall v = u+1, \ldots, i-1 \text{ and } v \neq 0 & \text{if } u = i + \lceil N/2 \rceil - 1 \text{ and } N \text{ odd}, \\ v, \forall v = u+1, \ldots, i-1 \text{ and } v \neq 0 & \text{if } u = i + N/2 - 1,\ i \neq 1 \text{ and } N \text{ even}, \\ v, \forall v = N/2+1, \ldots, N-1 & \text{if } u = 0,\ i = 1 \text{ and } N \text{ even}, \\ \emptyset & \text{otherwise.} \end{cases}$$

It is easy to verify that these trees are edge-disjoint and of maximal depth 3 (see Fig. 17). Thus following Proposition 4.8:

$$b_{H*}(K_N) \leqslant (\sqrt{L\tau/\lfloor N/2 \rfloor} + \sqrt{2\beta})^2.$$

For an even number of processors, the propagation time reaches the lower bound when the size of the messages is increasing.

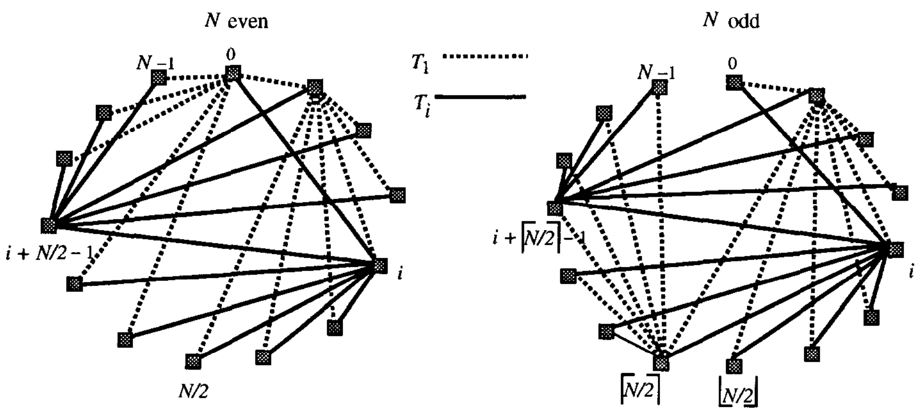

Fig. 17. Two disjoint spanning trees of K_N.

Berge [5] has shown that the chromatic index of K_N is $N-1$ if N is even, and N if N is odd. Therefore, applying Corollary 4.11 would imply a large amount of start-up time for the processor-bound model. Thus it is cheaper to use two hamiltonian paths, say P_0 and P_1. We label P_0 with odd numbers, and P_1 with even numbers. The delay is 4, and the maximum label is $2(N-1)$, hence following Proposition 4.10:

$$b_{H1}(K_N) \leqslant (\sqrt{2L\tau} + \sqrt{2(N-3)\beta})^2.$$

Ring graph C_N: We refer to [93] for the description of a pipelined broadcast:

$$b_{H*}(C_N) \leqslant (\sqrt{L\tau} + \sqrt{(\lfloor N/2 \rfloor - 1)\beta})^2.$$

We define a processor-bound labelling of the shortest path tree rooted, without loss of generality, at node 0 by $\mathrm{lab}(u-1,u) = u$ for $1 \leqslant u \leqslant \lfloor N/2 \rfloor$, and $\mathrm{lab}(u+1,u) = N-u+1$ for $\lfloor N/2 \rfloor + 1 \leqslant u \leqslant N-1$. The maximum label is $\lceil N/2 \rceil$ and the delay is $\varrho = 2$. Thus following Proposition 4.10, we get:

$$b_{H1}(C_N) \leqslant (\sqrt{2L\tau} + \sqrt{(\lceil N/2 \rceil - 2)\beta})^2.$$

2-Torus graph $T_N = C_p \square C_q$: We define two spanning trees T_1 and T_2 of $C_p \square C_q$. The rows are labelled from 0 to $p-1$, while the columns are labelled from 0 to $q-1$. Assume, without loss of generality, the source of the broadcasting to be $(0,0)$. The following describes the father function of T_1, T_2 being obtained by simple transposition:

$$\mathrm{father}((l,c),T_1) = \begin{cases} (0,c-1) & \text{if } l=0 \text{ and } c \neq 0, \\ (l-1,c) & \text{if } l \neq 0 \text{ and } c \neq 0, \\ (l,q-1) & \text{if } l \neq 0 \text{ and } c = 0. \end{cases}$$

Clearly these two trees are edge-disjoint and of maximal depth $p+q-1$. Thus following Proposition 4.8:

$$b_{H*}(C_p \square C_q) \leqslant (\sqrt{L\tau/2} + \sqrt{(p+q-2)\beta})^2.$$

This time is very close to the bound. Simmen [96] gave a similar result when $p=q$, and Bermond et al. [13] constructed two edge-disjoint spanning trees of maximal depth $D+2$.

Now, under the processor-bound model, following Lemma 4.13 and Corollary 4.11, we get:

$$b_{H1}(C_p \square C_q) \leqslant \begin{cases} (\sqrt{2L\tau} + \sqrt{3(p+q-1)\beta})^2 & \text{if } p \text{ or } q \text{ is even,} \\ (\sqrt{\tfrac{5}{2}L\tau} + \sqrt{4(p+q-1)\beta})^2 & \text{if } p \text{ and } q \text{ are odd.} \end{cases}$$

The exact complexity of the broadcast in $C_p \square C_q$ under the half-duplex processor-bound model for odd p and q is not known.

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: Similar considerations as in the full-duplex model allow to build d edge-disjoint spanning trees as follows: $T_1 = (e_1, \ldots, e_d)$, and

$T_i = R^i(T_1)$, $i = 2, \ldots, d$ where:

$$R = \begin{pmatrix} 0 & & & & & 1 \\ 1 & 0 & & & & 0 \\ 0 & 1 & . & & & 0 \\ & 0 & . & . & & . \\ & & . & . & 0 & 0 \\ & & & 0 & 1 & 0 \end{pmatrix}.$$

The depth of these trees is $\sum_{i=1}^{d} p_i - d + 1 \leqslant 2D + 1$, therefore we obtain:

$$b_{H*}(C_{p_1} \square \cdots \square C_{p_d}) \leqslant (\sqrt{L\tau/d} + \sqrt{2D\beta})^2,$$

and following again Lemma 4.13 and Corollary 4.11 under the processor-bound model, we get:

$$b_{H1}(C_{p_1} \square \cdots \square C_{p_d}) \leqslant \begin{cases} (\sqrt{2L\tau} + \sqrt{2D(2d-1)\beta})^2 & \text{if one of the } p_i\text{'s is even,} \\ \left(\sqrt{\dfrac{2d+1}{d} L\tau} + \sqrt{4dD\beta}\right)^2 & \text{if all the } p_i\text{'s are odd.} \end{cases}$$

In the particular case $d = 3$, Darrot [27] has built 3 edge-disjoint spanning trees of $C_{p_1} \square C_{p_2} \square C_{p_3}$ of maximum depth $D + \max_{i=1,2,3} \lceil p_i/2 \rceil + 2$.

The exact complexity of the broadcast in $C_{p_1} \square \cdots \square C_{p_d}$ under the full-duplex processor-bound model when all the p_i's are odd is not known.

Hypercube graph H_d: It is proved in [2], as a consequence of results from [3], that every d-cube, d even, has a Hamilton decomposition. More generally, there exist $\lfloor d/2 \rfloor$ edge-disjoint hamilton cycles in H_d. Applying Proposition 4.8, these cycles give a protocol for broadcasting:

$$b_{H*} \leqslant (\sqrt{L\tau/\lfloor d/2 \rfloor} + \sqrt{(N-2)\beta})^2.$$

However, the start-up times cannot be neglected except for very long messages because $N \gg D = \log_2 N$. Thus we are interested in finding edge-disjoint spanning trees of H_d of small depth. Ho showed that there exist $\lfloor d/2 \rfloor$ edge-disjoint spanning trees of maximum depth $3d/2 + 3$. Is it possible to do better?

Cube connected cycles graph CCC_d: Pipelining along a shortest paths tree (of depth $2d + \lfloor d/2 \rfloor - 2$ [79]) allows to complete a link-bound broadcasting, and thus

$$b_{H*}(CCC_d) \leqslant (\sqrt{L\tau} + \sqrt{(2d + \lfloor d/2 \rfloor - 3)\beta})^2.$$

Under the processor-bound model, it is easy to construct a processor-bound labeling (see Section 4.1.2) of a spanning tree of CCC_d of maximum label $2d + \lceil d/2 \rceil - 1$, thus Proposition 4.5 implies

$$b_{H1}(CCC_d) \leqslant (\sqrt{3L\tau} + \sqrt{(2d + \lceil d/2 \rceil - 4)\beta})^2.$$

The exact complexity of broadcasting is not known.

de Bruijn graph $UB(d, D)$: Here again, we refer to [7] where $d - 1$ EDST's of $UB(d, D)$ of maximum depth $D + 2\lfloor D/2 \rfloor + 1$ are described. Therefore:

$$b_{H*}(UB(d, D)) \leqslant (\sqrt{L\tau/(d-1)} + \sqrt{2D\beta})^2.$$

Following Lemma 4.15 and Corollary 4.11:

$$b_{H1}(UB(d, D)) \leqslant (\sqrt{2d/(d-1)L\tau} + \sqrt{2D(2d-1)\beta})^2.$$

Star graph S_n: Kundu's theorem [62] shows that there are $\lfloor (n-1)/2 \rfloor$ edge-disjoint spanning trees in the star graph S_n. However, we do not know any construction. For instance, S_n is hamiltonian [58], but we do not even know if there exists a decomposition of S_n in $\lfloor (n-1)/2 \rfloor$ edge-disjoint hamiltonian paths. Thus broadcasting in S_n under the half-duplex model is still an *open problem.*

4.2. Gossiping

Before proceeding with our discussion on gossiping in usual networks, we want to point out that studies on minimum gossip graphs under the linear model have been recently presented in [42].

We present below a general lower bound for gossiping:

Proposition 4.17. *Let G be a regular graph of degree Δ, then*

$$g_{F1}(G) \geqslant \begin{cases} (N-1)L\tau & \text{if } N \text{ even}, \\ N\,L\tau & \text{if } N \text{ odd}, \end{cases} \qquad g_{F*}(G) \geqslant (N-1)(L/\Delta)\tau,$$

$$g_{H1}(G) \geqslant \begin{cases} 2(N-1)L\tau & \text{if } N \text{ even}, \\ 2N\,L\tau & \text{if } N \text{ odd}, \end{cases} \qquad g_{H*}(G) \geqslant 2(N-1)(L/\Delta)\tau.$$

Proof. The technical terms of this proof are defined in Section 4.1.1. The bound under the model $F*$ is obtained using the root dominance. The bound under the model $F1$ and for the half-duplex models are obtained with the bandwidth dominance. Recall that the total available bandwidth under the half-duplex model is $N\Delta/(2\tau)$ if the nodes can communicate simultaneously over all their links, and $\lfloor N/2 \rfloor/\tau$ if only one link can be used at any given time by each processor. The total amount of exchanged messages is at least $N(N-1)L$ for a gossiping. $\square$

We generally do not know the amount of start-up times that we can add to the lower bounds on the propagation time. However, we can give lower bounds of communication costs as maxima of two terms: minimum start-up time and minimum propagation time (see [55, 99]). On the other hand, the best known algorithms often have a cost which is the sum of the minimum start-up time and the minimum propagation time, thus they are optimal within a multiplicative factor of two.

4.2.1. Full-duplex model

Under the full-duplex link-bound model, the simple *greedy algorithm* described in Section 3, performing in D steps in a network of diameter D, is not always efficient when the message size is large. Indeed, it does not allow to minimize the propagation time since the same information passing through distinct paths can be received many times by a same processor. Therefore, we shall give new algorithms for the link-bound model. Likewise, under the processor-bound model, algorithms of Section 3 minimize the number of steps, but sometimes to the prejudice of the propagation time under the linear model.

Complete graph K_N: Under the link-bound model $g_{F*}(K_N) = \beta + L\tau$. Under the processor-bound model, the complexity depends on the parity of N (see [42, 57]). For N even, the problem is solved:

$$g_{F1}(K_N) = \lceil \log_2 N \rceil \beta + (N-1)L\tau, \quad \text{if } N \text{ is even.}$$

For N odd, the complexity of $g_{F1}(K_N)$ is not known.

Ring graph C_N: Under the link-bound model, the greedy algorithm consists for every processor to send its message to its neighbours, which propagate in turn the messages. However, if N is even, we modify the last step in order for every node not to receive twice the message of the node farthest away (see [38]): the $N/2$-th step consists in sending only half of the last message received (for instance upper half in the clockwise direction, and the lower half in the counterclockwise direction).

$$g_{F*}(C_N) \leqslant \lfloor N/2 \rfloor \beta + (N-1)(L/2)\tau.$$

Under the processor bound model, if N is even we label the links with 0 and 1 alternatively. Each link is used during the steps equal to its label modulo 2:

$$g_{F1}(C_N) \leqslant \beta + L\tau + (N/2 - 1)(\beta + 2L\tau) = (N/2)\beta + (N-1)L\tau, \quad \text{if } N \text{ is even.}$$

For odd N, we can perform a gossiping in $(3N/4)\beta + (3N/2)L\tau$. However, this algorithm may not be optimal, and the exact complexity of the processor-bound gossiping in C_N is not known for N odd.

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: In $C_p \square C_p$, we can perform a gossip in two phases: first a horizontal gossip on each row, and next a vertical gossip on each column. It follows: $g_{F*}(C_p \square C_p) \leqslant \lfloor p/2 \rfloor \beta + (p-1)(L/2)\tau + \lfloor p/2 \rfloor \beta + (p-1)\, p(L/2)\tau = 2\lfloor p/2 \rfloor \beta + (N-1)(L/2)\tau$. However, the bandwidth is not completely used. We can do better: each message is cut in two blocks, block 1 and block 2; two gossips are simultaneously executed, first vertically and next horizontally on blocks 1, and another horizontally and next vertically on blocks 2. We obtain:

$$g_{F*}(C_p \square C_p) \leqslant 2\lfloor p/2 \rfloor \beta + (N-1)(L/4)\tau.$$

Fraigniaud presents in [38] a generalization of this approach for a link-bound gossiping in T_N where $p_1 = \cdots = p_d = p$, and $N = p^d$. It consists in splitting each message in d blocks and performing d phases of gossiping; each phase consisting in d simultaneous gossips in the d directions with one block per direction per node, the directions being rotated in the next phase:

$$g_{F*}(C_p \square \cdots \square C_p) \leqslant d\lfloor p/2 \rfloor \beta + \frac{(N-1)L}{2d}\tau.$$

We can define a processor-bound gossiping on the torus from the one defined before for the ring. If all the cycles orders are even, then we obtain:

$$g_{F1}(C_{p_1} \square \cdots \square C_{p_d}) \leqslant \sum_{i=1}^{d} \frac{p_i}{2}\beta + (N-1)L\tau \quad \text{if all the } p_i\text{'s are even.}$$

As far as we know, the complexity of gossiping in any torus is not known.

Hypercube graph H_d: The algorithm described in Section 3 performs by exchanging the sides of the hypercube. At each step, the message size is doubled, hence the total cost is $\sum_{i=0}^{d}(\beta + 2^i L\tau)$. Moreover, the processors use only one link at a time. Hence

$$g_{F1}(H_d) \leqslant (N-1)L\tau + d\beta.$$

Ho and Johnsson [55] and Stout and Wagar [99] proposed a gossip in the hypercube which consists in splitting the messages in d blocks and in performing simultaneously d versions of the above algorithm, each applied on each block. The ith version performs by exchanging the messages successively through dimensions $i, i+1, \ldots, d-1, 0, 1, \ldots, i-1$. We deduce:

$$g_{F*}(H_d) \leqslant (N-1)(L/d)\tau + d\beta.$$

Note that the propagation times of the two protocols presented above are both optimal. Moreover, it was conjectured by Ho [49] that these two algorithms are in fact totally optimal. The optimality has been proved in [42] for the processor-bound model.

Cube connected cycles graph CCC_d: Up to our knowledge, the problem of gossiping in CCC_d is still an *open problem.*

de Bruijn graph $UB(d, D)$: Bermond and Fraigniaud [7] have shown that the greedy algorithm is efficient in the de Bruijn network under the link-bound model. Indeed, it performs in less than $(N-1)/(d-1)L\tau + D\beta$ which minimizes the propagation time. Moreover, because of asymmetries in the load of the links during the execution of the gossip, they claim that it may be possible to find a faster algorithm. This would give a first example of gossiping performing in less than the minimum propagation time plus the minimum start-up time.

The complexity of a processor-bound gossiping in de Bruijn networks is still an *open problem.*

Star graph S_n: MacKenzie and Seidel [75] showed that

$$g_{F*}(S_n) \leqslant \frac{N-1}{n-1} L\tau + \lfloor 3(n-1)/2 \rfloor \beta.$$

The complexity of a processor-bound gossiping in S_n is still an *open problem.*

4.2.2. *Half-duplex model*

The bounds of Proposition 4.17 imply that if we know a gossiping algorithm under the full-duplex model which reaches the minimum propagation time, then we also know a gossiping algorithm under the half-duplex model which reaches the minimum propagation time. Indeed, the minimum propagation time under the half-duplex model is twice the one under the full-duplex model, and we can easily perform a full-duplex gossiping in twice its cost on a machine supporting only half-duplex communications. Of course such a method does not define optimal half-duplex gossiping, due to the amount of start-up times, but under the linear model, we are mainly interested in minimizing the propagation time. We refer to Section 3 for the description of half-duplex gossiping performing in a minimum start-up time.

Complete graph K_N: From Section 4.2.1, $g_{H*}(K_N) \leqslant 2(\beta + L\tau)$. Under the processor-bound model, the complexity depends on the parity of N (see Section 4.2.1).

Ring graph C_N: A half-duplex version of the greedy algorithm presented in Section 4.2.1 is given in [93]:

$$g_{H*}(C_N) \leqslant (N-1)(\beta + L\tau).$$

Under the processor-bound model, the complexity depends on the parity of N (see Section 4.2.1).

d-Torus graph $T_N = C_{p_1} \square \cdots \square C_{p_d}$: From Section 4.2.1,

$$g_{H*}(C_p \square C_p) \leqslant 2(p-1)\beta + (N-1)(L/2)\tau$$

which, for long messages, is about twice faster as the algorithm proposed in [93]. From Section 4.2.1, we obtain in general:

$$g_{H*}(C_p \square \cdots \square C_p) \leqslant d(p-1)\beta + \frac{(N-1)L}{d}\tau.$$

Under the processor-bound model, the complexity depends on the parity of the length of the cycles (see Section 4.2.1).

Hypercube graph H_d: From Section 4.2.1, $g_{H1}(H_d) \leqslant 2(N-1)L\tau + 2d\beta$, and $g_{H*}(H_d) \leqslant 2(N-1)(L/d)\tau + 2d\beta$. In both cases, the propagation times are optimal. However, it is possible to decrease the start-up times. For instance, Saad and Schultz [92] present a gossip whose cost is $2((N+d^2)/d)L\tau + (d+1)\beta$ under the link-bound model.

Cube connected cycles graph CCC_d: To our knowledge, the problem of gossiping in CCC_d is still an *open problem.*

de Bruijn graph $UB(d, D)$ *and Star graph* S_n: From Section 4.2.1, we obtain upper bounds under the link-bound model. However, the complexity of the processor-bound gossiping in $UB(d, D)$ and S_n is still an *open problem.*

4.3. Scattering and multiscattering

We present below general lower bounds for scattering and multiscattering

Proposition 4.18. *Let G be a regular graph of degree* Δ,

$$s_{F1}(G) \geqslant (N-1)L\tau, \quad s_{F*}(G) \geqslant (N-1)(L/\Delta)\tau,$$
$$s_{H1}(G) \geqslant (N-1)L\tau, \quad s_{H*}(G) \geqslant (N-1)(L/\Delta)\tau.$$

Proof. All bounds are obtained using the root dominance (see Section 4.1.1). □

Proposition 4.19. *Let* $|\Gamma_u^{(i)}|$ *be the number of nodes at distance i from a node u of a graph or digraph G. If* $m(G)$ *is the minimum time of a multiscattering of messages of same length L, then*

$$m(G) \geqslant \frac{(\sum_{u \in V} \sum_{i=1}^{D} i|\Gamma_u^{(i)}|)L}{B_G}$$

where B_G *is the total available bandwidth of the network, depending on the different models studied.*

Proof. The total bandwidth required for a multiscattering of messages of same length L is at least $(\sum_{i=1}^{D} i|\Gamma_u^{(i)}|)L$ for each node u of the network. The result follows using the bandwidth dominance (see Section 4.1.1). □

Except in the case of the hypercube, few results have been given about the complexity of the personalized communications under the linear model. Therefore, we present only a short list of results for a restricted number of topologies. Up to our knowledge, the scattering and multiscattering problems on the other usual topologies have not been yet studied.

Ring graph C_N: A scattering under the link-bound models performs in $\lfloor N/2 \rfloor$ steps by sending, at step i, messages destinated to vertices at distance $\lfloor N/2 \rfloor - i + 1$. If N is even, we modify the last step as in the gossiping: node $N/2$ is reached by two different directions. This decreases the propagation time, and rather improves the result of [93]:

$$s_{F*}(C_N) \leqslant s_{H*}(C_N) \leqslant \lfloor N/2 \rfloor \beta + \frac{N-1}{2} L\tau.$$

Under the processor-bound model, it is easy to show that:

$$s_{F1}(C_N) \leqslant s_{H1}(C_N) \leqslant (N-1)(\beta + L\tau).$$

Note that Fraigniaud et al. [41] have shown that the complexity of scattering on an *oriented* ring of processors is $s(\vec{C}_N) = (N-1)(\beta + L\tau)$ under the link-bound mode. Regarding this problem, see also [18] and [74b].

The multiscattering under the $F*$ model can be done in executing simultaneous scatterings from every node and grouping messages: during the first round, 1 message is exchanged between each couple of neighbours, during the second round, 2 messages are exchanged, during the third round, 3 messages are exchanged, and so on. Therefore:

$$m_{F*}(C_N) \leqslant \lfloor N/2 \rfloor \beta + \begin{cases} \dfrac{N^2-1}{8} L\tau & \text{if } N \text{ is odd,} \\ \dfrac{N^2}{8} L\tau & \text{if } N \text{ is even.} \end{cases}$$

On a machine supporting only half-duplex communication, the cost is doubled. This shows that it is possible to multiscatter in a propagation time less than $(N^2/4)L\tau$ which is twice faster than what is obtained in [93] with a direct algorithm under the link-bound model.

Torus graph $C_{\sqrt{N}} \square C_{\sqrt{N}}$: The algorithms presented in [38] use the same idea than the one presented for the gossiping: the use of all 4 directions simultaneously to perform scattering or multiscattering on messages of length $L/2$ on each direction using the algorithms developed for the ring. This gives:

$$s_{F*}(C_{\sqrt{N}} \square C_{\sqrt{N}}) \leqslant 2\lfloor \sqrt{N}/2 \rfloor \beta + (N-1)(L/4)\tau$$

and

$$m_{F*}(C_{\sqrt{N}} \square C_{\sqrt{N}}) \leqslant \begin{cases} (\sqrt{N}-1)\beta + \dfrac{(N-1)\sqrt{N}}{8} L\tau & \text{if } \sqrt{N} \text{ is odd,} \\ \sqrt{N}\beta + \dfrac{N\sqrt{N}}{8} L\tau & \text{if } \sqrt{N} \text{ is even.} \end{cases}$$

The half-duplex versions of these algorithms are much more efficient than the ones proposed in [93].

This approach can be generalized to the d-torus, $d \geqslant 1$, $C_p \square \cdots \square C_p$, with $N = p^d$. We obtain $s_{F*}(C_p \square \cdots \square C_p) \leqslant d\lfloor p/2 \rfloor \beta + (N-1)L/(2d)\tau$ and

$$m_{F*}(C_p \square \cdots \square C_p) \leqslant \begin{cases} d\dfrac{p-1}{2}\beta + \dfrac{(p^2-1)N}{8p}L\tau & \text{if } p \text{ is odd,} \\ d\dfrac{p}{2}\beta + \dfrac{pN}{8}L\tau & \text{if } p \text{ is even.} \end{cases}$$

Results for the half-duplex model follow.

Concerning the processor-bound model:

$$s_{F1}(C_{\sqrt{N}} \square C_{\sqrt{N}}) \leqslant s_{H1}(C_{\sqrt{N}} \square C_{\sqrt{N}}) \leqslant 2(\sqrt{N}-1)\beta + (N-1)L\tau.$$

Hypercube graph H_d: The main result is due to Stout and Wagar [99] who proved that

$$s_{F*}(H_d) \leqslant \frac{d}{\log d}\beta + \frac{N-1}{d}L\tau.$$

Concerning the processor-bound model, Ho and Johnsson [55] proved that $s_{F1}(H_d) \leqslant d\beta + (N-1)L\tau$ using the spanning binomial tree (SBT) algorithm. This applies also for the half-duplex model. Finally using rotations of the SBT tree (d-Rotated-SBT), Ho and Johnsson [55] showed that $s_{H*}(H_d) \leqslant d\beta + ((N-1)/d)L\tau$. Since this approach needs to split messages, they also proposed algorithms based on balanced spanning trees [51] (see also [17]).

The SBT, the dRSBT, and the balanced spanning tree can also be used for multiscattering [55, 99]: $m_{F1}(H_d) \leqslant d\beta + (dN/2)L\tau$ and $m_{F*}(H_d) \leqslant d\beta + (N/2)L\tau$.

Cube connected cycles graph CCC_d: Li [72] has studied the complexity of the multiscattering on reconfigurable topology: $m_{F*}(CCC_d) \leqslant (3d-1)(\beta + (N/2)L\tau)$ and $m_{H*}(CCC_d) \leqslant (4d-1)(\beta + (N/2)L\tau)$.

5. Conclusion

We presented the known results about communication in usual networks, but hopefully there are still a lot of open problems to be solved, or improvements to be made. We list them here in a concise form. We shall use the following notations:

- D: (Done) The exact value is known or tight bounds can be given.
- TI: (To Improve) Results are to be improved in a significant way.
- ?: Open problem.

This classification is somehow arbitrary. We shall say that results are to be improved when nontrivial results have already been found but where we think that some further work might improve the bounds.

Table 2 presents the results for the broadcasting under the constant model. We only give the $F1$ results because the other ones are straightforward or direct consequences of $F1$.

Table 3 presents the results for the gossiping under the constant model. The $F*$ model is not given since it is straightforward.

Table 4 presents the results for the broadcasting under the linear model. As the $F*$ model has been completely solved (excepted sometime what is concerning the depth of the disjoint spanning trees used for the algorithms, and the local definition of these trees), it is not given in the table. Note that the broadcast problem has not been studied for the grids, the shuffle-exchange and the butterfly graphs.

Table 5 presents the results for the gossiping under the linear model. The half-duplex models are not given as their times are just twice the full-duplex times. As for the broadcasting, the grids, shuffle-exchange and butterfly graphs have not been studied.

Using constant time, in the DMA-bound models, few results are to be improved for broadcasting, but for the half-duplex gossiping everything has to be done. Using linear time, nearly nothing has been done with the DMA-bound models, and only a few results are known for scattering and multiscattering. Also, in most of the cases, the complexities of the problems are only known for large messages.

Table 2
Broadcasting under the constant model

	K_N	C_N	$C_p \square C_q$	$C_{p_1} \square \cdots \square C_{p_d}$	$P_{n_1} \square \cdots \square P_{n_d}$	H_d	CCC_d	$UB(d, D)$	S_n	SE_d	BF_d
$F1$	D	D	D	TI	D	D	D	TI	TI	D	TI

Table 3
Gossiping under the constant model

	K_N	C_N	$C_p \square C_q$	$C_{p_1} \square \cdots \square C_{p_d}$	$P_{n_1} \square \cdots \square P_{n_d}$	H_d	CCC_d	$UB(d, D)$	S_n	SE_d	BF_d
$H*$	D	D	TI	TI	TI	D	TI	D	D	TI	TI
$F1$	D	D	D	TI	D	D	TI	?	TI	?	TI
$H1$	D	D	TI	TI	TI	TI	TI	?	?	?	TI

Table 4
Broadcasting under the linear model

	K_N	C_N	$C_p \square C_q$	$C_{p_1} \square \cdots \square C_{p_d}$	H_d	CCC_d	$UB(d, D)$	S_n
$F1$	D	TI	TI	TI	D	D	TI	D
$H*$	D	D	D	D	?	TI	D	?
$H1$	D	D	TI	TI	?	TI	D	?

Table 5
Gossiping under the linear model

	K_N	C_N	$C_p \square C_q$	$C_{p_1} \square \cdots \square C_{pd}$	H_d	CCC_d	$UB(d, D)$	S_n
$F*$	D	D	TI	TI	D	?	D	?
$F1$	TI	TI	TI	TI	D	?	?	?

Acknowledgement

Both authors are grateful to Dominique Sotteau and one of the referees for many helpful remarks.

References

[1] S. Akers, D. Harel and B. Krishnamurthy, The star graph: an attractive alternative to the n-cube, in: Proceedings of the International Conference on Parallel Processing, PA (1987) 393–400.

[2] B. Alspach, J.-C. Bermond and D. Sotteau, Decomposition into cycles 1: Hamilton decompositions, in: G. Hahn et al., eds., Cycles and Rays (Academic Press, New York, 1990) 9–18.

[3] J. Aubert and B. Schneider, Décomposition de la somme cartésienne d'un cycle et de l'union de deux cycles hamiltoniens en cycles hamiltoniens, Discrete Math. 38 (1982) 7–16.

[4] B. Awerbuch, O. Goldreich, D. Peleg and R. Vainish, A tradeoff between information and communication in broadcast protocols, Tech. Rept. MIT/LCS/TM-365, Institute of Technology, MA (1988); J. ACM, to appear.

[4a] A. Ben-Dor, S. Halevi and A. Schuster, On greedy hot-potatoe routing, Tech. Rept. CS-LPCR 9302, Technion, Israel (1993).

[4b] A. Bagchi, S. Hakimi and E. Schmeichel, Gossiping in a distributed network, IEEE Trans. Comput. 42 (1993).

[4c] A. Bagchi and S. Hakimi, Data transfers in broadcast networks, IEEE Trans. Comput. 41 (1992).

[4d] R. Ber-Yehuda, A. Israeli and A. Itai, Multiple communication in multihop radio networks, SIAM J. Comput. 22 (1993) 875–887.

[5] C. Berge, Graphes (Gauthier-Villars, Paris, 1983).

[6] J.-C. Bermond, C. Delorme and J.-J. Quisquater, Strategies for interconnection networks: some methods from graph theory, J. Parallel Distributed Comput. 3 (1986) 433–449.

[7] J.-C. Bermond and P. Fraigniaud, Broadcasting and gossiping in de Bruijn networks, SIAM J. Comput. 23 (1994) 212–225.

[8] J.-C. Bermond and P. Fraigniaud, Communications in interconnection networks, in: Proceedings Combinatorial Optimization in Science and Technology '91 (1991).

[9] J.-C. Bermond and P. Fraigniaud, Broadcasting and NP-completeness, Graph Theory Notes of New York XXII (1992) 8–14.

[10] J.-C. Bermond, P. Fraigniaud and J. Peters, Antepenultimate broadcasting, Tech. Rept. CMPT TR 92-03, School of Computing Science, Simon Fraser University, Burnaby, B.C. (1992); also: Networks, to appear.

[11] J.-C. Bermond and P. Hell, Chromatic index of the Kautz and the de Bruijn digraphs, Tech. Rept., School of Computing Science, Simon Fraser University, Burnaby, B.C. (1990).

[12] J.-C. Bermond, P. Hell, A.L. Liestman and J.G. Peters, Broadcasting in bounded degree graphs, SIAM J. Discrete Math. 5 (1992) 10–24.

[12a] J-C. Bermond, J-C. Konig and M. Raynal, General and efficient decentralised consensus protocols, in: Second International Workshop on Distributed Algorithms, Amsterdam (1987), Lecture Notes in Computer Science 312 (Springer, Berlin, 1988) 41–56.

[13] J.-C. Bermond, P. Michallon and D. Trystram, Broadcasting on wraparound meshes with parallel monodirectional links, Parallel Comput. 18 (1992) 639–648.
[14] J.-C. Bermond and C. Peyrat, de Bruijn and Kautz networks: a competitor for the hypercube?, in: F. Andre and J.P. verjus, eds., Hypercube and Distributed Computers (North-Holland, Amsterdam, 1989) 279–294.
[15] J.-C. Bermond and C. Peyrat, Broadcasting in de Bruijn networks, in: Proceedings 19th S-E Conference Combinatorics, Graph Theory, and Computing, FL Congressus Numerantium 66 (1988) 283–292.
[16] P. Berthomé, A. Ferreira and S. Perennes, Tech. Rept., LIP, ENS-Lyon, France (1992).
[17] D.P. Bertsekas, C. Ozveren, G.D. Stamoulis, P. Tseng and J.N. Tsitsiklis, Optimal communication algorithms for hypercube, J. Parallel Distributed Comput. 11 (1991) 263–275.
[17a] S. Bhatt, G. Bilardi, G. Pucci, A. Ranade, A. Rosenberg and E. Schwabe, On bufferless routing of variable length messages in leveled networks, Tech. Rept. 93-18, University of Amherst, MA (1993).
[18] S. Bhatt, G. Pucci, A. Ranada and A. Rosenberg, Scattering and gathering messages in networks of processors, IEEE Trans. Comput. 42 (1993) 938–949.
[19] G.E. Blelloch, Scan as a primitive parallel operation, IEEE Trans. Comput. 38 (1989) 1526–1538.
[20] L. Bomans and D. Roose, Communication benchmarks for the iPSC/2, in: Hypercube and Distributed Computers (North-Holland, Amsterdam, 1989) 311–328.
[21] J. Bruck, R. Cypher and C-T. Ho, Multiple message broadcasting with generalized fibonacci trees, Tech. Rept. RJ 8763, IBM Almaden Research Center (1992).
[22] R.M. Capocelli, L. Gargano and U. Vaccaro, Time bounds for broadcasting in bounded degree graphs, in: WG 89: 15th International Workshop on Graphtheoretic Concepts in Computer Science, Rolduc (NL) (1989) 19–33.
[23] T. Champion and B. Tourancheau, Tnode: document utilisateur, Tech. Rept., LIP-IMAG 89-02, ENS Lyon, France (1989).
[24] M. Cosnard and P. Fraigniaud, Finding the roots of a polynomial on an MIMD multicomputer, Parallel Comput. 15 (1990) 75–85.
[25] G. Cybenko, D.W. Krumme and K.N. Venkataraman, Gossiping in minimum time, SIAM J. Comput. 21 (1992) 111–139.
[26] W.J. Dally, Performance analysis of k-ray n-cube interconnection networks, IEEE Trans. Comput. 39 (1990) 775–785.
[26a] W. Dally and C. Seitz, Deadlock-free message routing in multiprocessor interconnection networks, IEEE Trans. Comput. 36 (1987) 547–553.
[27] E. Darrot, Manuscript.
[28] S. Djelloul, Etudes de certains réseaux d'interconnexion: structures et communications, PhD Thesis, Université Paris-Sud, Orsay (1992).
[28a] J. Duato, A new theory of deadlock-free adaptive routing in wormhole networks, IEEE Trans. Parallel Distrib. Systems 4 (1993) 1320–1331.
[29] A. Edelman, Optimal matrix transposition and bit reversal on hypercubes: All to all personalized communication, J. Parallel Distributed Comput. 11 (1991) 328–331.
[30] J. Edmonds, Edges-disjoint branchings, combinatorial algorithms, in: R. Rustin, ed., Combinatorial Algorithms (Algorithmics press, New York, 1972) 91–96.
[31] R.C. Entringer and P.J. Slater, Gossips and telegraphs, J. Franklin Inst. 307 (1979) 353–359.
[32] S. Even and B. Monien, On the number of rounds necessary to disseminate information, ACM Symposium on Parallel Algorithms and Architectures, Santa Fe, June 1989.
[33] V. Fack, K. Coolsaet and H. De Meyer, Optimal algorithms for total exchange without buffering on the hypercube, BIT 32 (1992) 559–569.
[34] A. Farley and A. Proskurowski, Gossiping in grid graphs, J. Combin. Inform. Systems Sci. 5 (1980) 161–172.
[35] A. Farley and A. Proskurowski, Broadcasting: Between whispering and shouting, Tech. Rept. CIS-TR-89-06, Department of Computer Science, Univesity of Oregon, Eugene, OR (1989).
[36] A. Farley and A. Proskurowski, Bounded-call broadcasting. Tech. Rept. CIS-TR-91-22, Department of Computer Science, Univesity of Oregon, Eugene, OR (1991).
[37] A.M. Farley and S.T. Hedetniemi, Broadcasting in grid graphs, in: Proceedings 9th S-E Conference Combinatorics, Graph Theory, and Computing, FL Congressus Numerantium XXI (1978) 275–288.

[37a] E. Fleury and P. Fraigniaud, Deadlocks in adaptive wormhole routing, Res. Rept. 94-09, LIP ENS-Lyon (1994).

[37b] E. Fleury and P. Fraigniaud, Stategies for multicasting in meshes, Res. Rept. 94-07, LIP ENS-Lyon (1994).

[37c] P. Fragopoulou and S. Akl, Optimal communication algorithms on star graphs using spanning tree constructions, Tech. Rept. 93-346, Queen's University, Kingston, Ont. (1993).

[38] P. Fraigniaud, Communications intensives dans les architectures à mémoire distribuée et algorithmes parallèles de recherche de racines de polynômes, PhD Thesis, ENS Lyon (1990).

[39] P. Fraigniaud, Performance analysis of broadcasting in hypercubes with restricted communication capabilities, J. Parallel Distributed Comput. 16 (1992) 15–26.

[39a] P. Fraigniaud and C. Gavoille, Interval routing schemes, Res. Rept. 94-04, LIP ENS-Lyon (1994).

[39b] P. Fraigniaud and H-P. Charles, Scheduling a scattering gathering sequence on hypercubes, Parallel Process. Lett. 3 (1993) 29–42.

[40] P. Fraigniaud and C.-T. Ho, Arc-disjoint spanning trees on cube connected cycles networks, in: Proceedings of International Conference on Parallel Processing (1991) I225–I229.

[40a] P. Fraigniaud and C. Laforest, Disjoint spanning trees of small depth, in: Proceedings of ParCo '93, to appear.

[41] P. Fraigniaud, S. Miguet and Y. Robert, Scattering on a ring of processors, Parallel Comput. 13 (1990) 377–383.

[42] P. Fraigniaud and J. Peters, Optimal gossip graphs, Tech. Rept., School of Computing Science, Simon Fraser University, Burnaby, B.C. (1992).

[42a] P. Fraigniaud and J. Peters, Structured communication in torus networks, Res. Rept., SFU, Vancouver (1992).

[42b] S. Fujita and M. Yamashita, Fast gossiping on square mesh computer, Inform. Process. Lett. 48 (1993) 127–130.

[43] C. GowriSankaran, Broadcasting on recursively decomposable Cayley graphs, Discrete Appl. Math. 53 (1994) 171–182.

[43a] M. Grammatikakis, F. Hsu and M. Kraetzl, Multicomputer routing, Tech. Rept. 5/94, Curtin University of Technology, Australia (1994).

[43b] S. Graham and S. Seidel, The cost of broadcasting on Star graphs and k-ary hypercubes, IEEE Trans. Comput. 42 (1993).

[44] A. Hajnal, E.C. Milner and E. Szemeredi, A cure for the telephone disease, Canad. Math. Bull. 18 (1972) 447–450.

[45] S.M. Hedetniemi, S.T. Hedetniemi and A.L. Liestman, A survey of gossiping and broadcasting in communication networks, Networks 18 (1986) 319–349.

[46] M.C. Heydemann, J. Opatrny and D. Sotteau, Broadcasting and spanning trees in de Bruijn and Kautz networks, Discrete Appl. Math. 37/38 (1992) 297–317.

[47] W.D. Hillis, The Connection Machine (MIT Press, Cambridge, MA, 1985).

[47a] A. Hily, Algorithmes déchange d'information dans les réseaux à bus, Rapport de DEA, LRI Université de Paris-Sud (1993).

[48] C.T. Ho, Full bandwidth communications on folded hypercubes, in: Proceedings International Conference on Parallel Processing (1990) I276–I280.

[49] C.T. Ho, Optimal communication primitives and graphs embeddings on hypercubes, PhD Thesis, Yale University, New Haven, CT (1990).

[50] C.T. Ho, Optimal broadcasting on SIMD hypercubes without indirect addressing capability, J. Parallel Distributed Comput. 13 (1991) 246–255.

[51] C.T. Ho and S.L. Johnsson, Spanning balanced trees in boolean cubes, Tech. Rept. YALEU/DCS/TR-508, Yale University, New Haven, CT (1987).

[51a] C-T. Ho, Matrix transpose on meshes with wormhole and XY routing, Res. Rept. RJ 9385 (82637), IBM, San Jose (1993).

[51b] C-T. Ho and M-Y. Kao, Optimal broadcast on hypercubes with wormhole and E-cube routing, Res. Rept. RJ 9394 (82646), IBM, San Jose (1993).

[52] C.T. Ho and M.T. Raghunath, Efficient communication primitives on hypercubes, Tech. Rept. RJ (72915), IBM San Jose (1991).

[53] J. Hromkovic, C.D. Jeschke and B. Monien, Optimal algorithms for dissemination of information in some interconnection networks, Tech. Rept. no. 71, Universitat Paderborn (1990).
[53a] J. Hromkovic, R. Klasing, B. Monien and R. Peine, Dissemination of information in interconnection networks (broadcasting and gossiping), Tech. Rept., University of Paderborn (1993).
[54] D. Johnson and M. Garey, Computers and Intractability: A Guide to the Theory of NP-Completeness (Freeman, San Francisco, CA, 1979).
[55] S.L. Johnsson and C.T. Ho, Optimum broadcasting and personalized communication in hypercubes, IEEE Trans. Comput. 38 (1989) 1249–1268.
[56] R. Klasing, B. Monien, R. Peine and E. Stöhr, Broadcasting in butterfly and deBruijn networks, Discrete Appl. Math. 53 (1994) 183–197.
[57] W. Knodel, New gossips and telephones, Discrete Math. 13 (1975) 95.
[58] V.L. Kompel'makher and V.A. Liskovets, Sequential generation of arrangements by mean of a basis of transpositions, Kibernetica 3 (1975) 17–21.
[59] J.C. Konig, D.W. Krumme and E. Lazard, Oriented torus diameter, Networks, to appear.
[60] J.-C. König and E. Lazard, Minimum k-broadcast graphs, Discrete Appl. Math. 53 (1994) 199–209.
[61] D.W. Krumme, Fast gossiping for the hypercube, SIAM J. Comput. 21 (1992) 365–380.
[62] S. Kundu, Bounds on the number of disjoint spanning trees, J. Combin. Theory 17 (1974) 199–203.
[63] S. Kuppuswami and B. Tourancheau, Evaluating the performances of transputer based hypercube vector computer, La Lettre Transput. 4 (1990) 23–33.
[64] R. Labahn, Some minimum gossip graphs, Tech. Rept. 91708-OR, University of Bonn (1991).
[64a] R. Labhan, Information flows on hypergraphs, Discrete Math. 113 (1993) 71–97.
[65] R. Labahn and C. Pietsh, Characterizing minimum gossip graphs on 16 vertices, Tech. Rept. 91707-OR, University of Bonn (1991).
[66] R. Labahn and I. Warnke, Quick gossiping by telegraphs, Discrete Math., to appear.
[67] R. Labahn and I. Warnke, Quick gossiping by multi-telegraphs, in: R. Bodendiek and R. Henn, eds., Topics in Combinatorics and Graph Theory (Physica-Verlag, Heidelberg, 1990) 451–458.
[68] C. Laforest, Arbres de recouvrement arc-disjoints de hauteur minimum, Master's thesis, ENS-Lyon (1992).
[69] Y. Lan, A.-H. Esfahanian and L.M. Ni, Multicast in hypercube multiprocessors, J. Parallel Distributed Comput. 8 (1990) 30–41.
[70] S. Latifi and A. El-Amawy, On folded hypercubes, in: 1989 International Conference on Parallel Processing I, Penn State (1989) 180–187.
[71] E. Lazard, Broadcasting in DMA-bound bounded degree graphs, Discrete Appl. Math. 37/38 (1992) 387–400.
[71a] T. Leighton, Introduction to Parallel Algorithms and Architectures: Array, Trees, Hypercubes (Morgan Kaufman, Los Altos, CA, 1992).
[72] J.J. Li, Multiscattering on a reconfigurable network of processors, Tech. Rept. 91–03, LIP, ENS-Lyon (1991).
[73] A.L. Liestman and J.G. Peters, Broadcast networks of bounded degree, SIAM J. Discrete Math. 1 (1988) 531–540.
[74] A.L. Liestman and J.G. Peters, Minimum broadcast digraphs, Discrete Appl. Math. 37/38 (1992) 401–419.
[74a] A. Liestman and D. Richards, Network communication in edge-colored graphs: gossiping, IEEE Trans. Parallel Distrib. Systems 4 (1993).
[74aa] A. Liestman and D. Richards, Perpetual gossiping, Parallel Process. Lett., to appear.
[74b] Y-D. Lyuu, On the furthest-distance-first principle for data scattering with set-up, in: Proceedings of the 5th IEEE Symposium on Parallel and Distributed Processing (1993) 633–640.
[74c] Ph. MacKenzie, A lower bound for order preserving broadcast in the postal model, Parallel Process. Lett., to appear.
[75] D. MacKenzie and S. Seidel, Broadcasting on three multiprocessor interconnection topologies, Tech. Rept. CS-TR-89-01, Michigan Technological University, Houghton, MI (1989).
[76] M. Mahéo and J.-F. Saclé, Note on the problem of gossiping in multidimensional grids, Discrete Appl. Math. 53 (1994) 287–290.

[77] D. May, The next generation transputers and beyond, in: A. Bode, ed., Distributed Memory Computing, Lecture Notes in Computer Science 487 (Springer, Berlin, 1991) 7–22.

[78] J.E. McCanna, Orientations of the n-cube with minimum diameter, Discrete Math. 68 (1988) 309–310.

[79] D.S. Meliksetian and C.Y.R. Chen, Communication aspects of the cube-connected cycles, in: Proceedings of the International Conference on Parallel Processing (1990) I579–I580.

[80] D.S. Meliksetian and C.Y.R. Chen, Optimal routing algorithm and other communication issues of the cube-connected cycles, Tech. Rept., Syracuse University, Syracuse, NY (1990).

[81] V.E. Mendia and D. Sarkar, Optimal broadcasting on the star graph, IEEE Trans. Parallel Distrib. Systems 3 (1992) 389–396.

[82] P. Michallon, D. Trystram and G. Villar, Optimal broadcasting algorithms on torus, RR 872-I-01-92 LMC-IMAG, Grenoble (1992).

[82a] M. Middendorf, Minimum broadcast time is NP-complete for 3-regular planar graphs and deadline 2, Inform. Process. Lett. 46 (1993) 281–287.

[83] S. Miguet, Programmation dynamique et traitement d'images sur machines parallèles a mémoire distribuée, PhD Thesis, ENS-Lyon (1990).

[84] M. Missakian and M. Quenot, Architectures parallèles pour l'exécution d'algorithmes de la vision, in: Proceedings Journée Thématique sur la Conception et la Programmation des Machines Parallèles, Direction Générale de l'Armement-DRET, France (1991).

[85] D. Nassimi and S.S. Sahni, Data broadcasting in SIMD computers, IEEE Trans. Comput. 30 (1981) 101–107.

[85a] L. Ni and P. McKinley, A survey of wormhole routing techniques in direct networks, Computers 2(26) (1993) 62–76.

[85b] D. Peleg and J. Ullman, An optimal synchronizer for the hypercube, SIAM J. Comput. 18 (1989) 740–747.

[85c] D. Peters and J. Peters, Bounded depth broadcasting, Tech. Rept. CMPT TR 93-10, SFU, Vancouver (1993).

[85d] B. Plateau and D. Trystram, Optimal total exchange for a 3-D torus, Inform. Process. Lett. 42 (1992) 95–102.

[86] J. Peters and M. Syska, Broadcasting in a torus network with wormhole routing, Tech. Rept. CMPT TR 93-04, SFU, Vancouver (1993).

[87] F. Preparata and J. Vuillemin, The cube connected cycles: a versatile network for parallel computation, Comm. ACM 24 (1981) 300–309.

[87a] P. Raghavan and E. Upfal, Efficient routing in all-optical networks, in: Proceedings of STOC '94, to appear.

[88] J. Rattner, The new age of supercomputing, in: A. Bode, ed., Distributed Memory Computing, Lecture Notes in Computer Science 487 (Springer, Berlin, 1991) 1–6.

[89] F.S. Roberts and Y. Xu, On the optimal strongly connected orientations of city street graphs III: three east-west avenues or north-south streets, Tech. Rept. RRR 21-86, Rutgers University, New Brunswick, NJ (1986).

[90] F.S. Roberts and Y. Xu, On the optimal strongly connected orientations of city street graphs I: large grids, SIAM J. Discrete Math. 1 (1988) 199–222.

[91] F.S. Roberts and Y. Xu, On the optimal strongly connected orientations of city street graphs II: two east-west avenues or north-south streets, Networks 19 (1989) 221–233.

[91a] J. de Rumeur, Communication dans les Réseaux d'Interconnexion (Masson, Paris, 1994).

[92] Y. Saad and M.H. Schultz, Data communication in hypercubes, J. Parallel Distributed Comput. 6 (1989) 115–135.

[93] Y. Saad and M.H. Schultz, Data communication in parallel architectures, Parallel Comput. 11 (1989) 131–150.

[93a] S. Seidel, Broadcasting on linear arrays and meshes, Tech. Rept. ORNL/TM-12356, Oak Ridge, TN (1993).

[94] S.R. Seidel, Circuit-switched vs. store and forward solutions to symmetric communication problems, in: Proceedings of the 4th Conference on Hypercube Concurrent Computers and Application (1989).

[95] S.R. Seidel, M.H. Lee and S. Fotedar, Concurrent bidirectional communication on the Intel iPSC860 and iPSC2, CS-TR 90-06, Michigan Technological University, Houghton, MI (1990).

[95a] J-P. Sheu, W-H. Liaw and T-S. Chen, A broadcast algorithm in star graph interconnection networks, Inform. Process. Lett. 48 (1993) 237–241.
[96] M. Simmen, Comments on broadcast algorithms for two-dimensional grids, Parallel Comput. 17 (1991) 109–112.
[96a] C. Spencer, Circuit-switched structured communication on toroidal meshes, Master thesis, SFU, Vancouver (1994).
[96b] G. Stamoulis and J. Tsitsiklis, An efficient algorithm for multiple simultaneous broadcasts in the hypercube, Inform. Process. Lett. 46 (1993) 219–224.
[97] E. Stohr, Broadcasting in the butterfly network, Inform. Process. Lett. 39 (1991) 41–43.
[98] E. Stohr, On the broadcast time of the butterfly network, in: WG 91: 17th International Workshop on Graphtheoretic Concepts in Computer Science (1991) 226–229.
[99] Q.F. Stout and B. Wagar, Intensive hypercube communication, prearranged communication in link-bound machines, J. Parallel Distributed Comput. 10 (1990) 167–181.
[100] V.S. Sunderam and P. Winkler, Fast information sharing in a distributed system, Tech. Rept., Emory University, Atlanta, GA (1989).
[101] J-T. Tien, C.T. Ho and W.-P. Yang, Broadcasting on incomplete hypercubes, IEEE Trans. Comput. 42 (1993) 1393–1398.
[102] A. Touzene and B. Plateau, Optimal multinode broadcast on a mesh connected graph with reduced bufferization, in: A. Bode, ed., Distributed Memory Computing, Lecture Notes in Computer Science 487 (Springer, Berlin, 1991) 143–152.
[103] E. Varvarigos and D. Bertsekas, Multinode broadcast in hypercubes and rings with randomly distributed length of packets, IEEE Trans. Parallel Distrib. Systems 4 (1993).
[103a] E. Varvarigos and D. Bertsekas, Communication algorithms for isotropic tasks in hypercube and wraparound meshes, Parallel Comput. 18 (1992) 1233–1257.
[103b] O. Wolfson and A. Segall, The communication complexity of atomic commitment and of gossiping, SIAM J. Comput. 20 (1991) 423–450.