# 12

# MODELS FOR OPTICALLY INTERCONNECTED NETWORKS

**Pascal Berthomé and Michel Syska***

*LRI, Université Paris-Sud, Orsay, France*

**SlooP: joint project I3S-CNRS, INRIA and University of Nice - Sophia Antipolis, France*

## ABSTRACT

Switching techniques used in optically interconnected networks differ from those used in classical electronically interconnected networks. This yields new communication models. The aim of this chapter is to survey the results of communication models in three fields: the design of networks, the algorithmics of data communication and the computational models of multiprocessor systems interconnected with optical networks.

## 1 INTRODUCTION

Massively parallel computers are proposed as the solution for high performance computing. However, parallel computing involves a lot of data communications between the processors that cooperate on the same computation. The amount of time required to perform those communications is prohibitive to the overall performance of the systems considered. As a consequence, dense interconnection network design and fast collective communication protocols are the keys for achieving expected performances. Indeed, multiprocessors systems are made of independent processing units - equipped with a local memory - exchanging data over an interconnection network. Two kinds of topologies are used: point-to-point and multi-stage interconnection networks. For instance, hypercubes and grids are popular point-to-point interconnection networks used in parallel computers. Multi-stage interconnection networks were designed in the case of telecommunication networks but are also relevant for workstation based computing. In this case, a central switch provides a virtual complete

graph topology. Moreover, as the existence of standard message passing libraries such as PVM and MPI makes it easier today to program such systems, interconnections are getting more and more importance.

The impact of the optical technology on the network modeling is investigated in the following. Three types of models are given: interconnection models (topologies), communication models and computational models.

Usually a $N$ nodes multiprocessor system is represented by a graph $G = (V, E)$, in which $V$ is the set of nodes of the graph representing the processors (and local memory associated to that processor) and $E$ is the set of edges of the graph representing the communication links between processors. The model is accurate for parallel computers where processors are pair-wise connected, each edge representing the connection between two neighbors. The communication of data between two nodes is thus of the one-to-one type. In the case of optical interconnection networks, it is "easy" to implement a one-to-many type of communication, extending the concept of neighbors. This can be represented by hypergraphs and will be developed in Section 3.

When one node has a piece of data to communicate to other nodes, the corresponding message in which the data is encapsulated may have to switch through intermediate nodes, thus introducing delay in the time required to deliver the message to its destination node. Collective communications corresponds to the case when the communication implies more than two nodes.

Two paradigms of elementary collective communications are usually considered: *broadcasting* and *gossiping*. In broadcasting, one node has a piece of data it would like to share with all the other nodes in the network. At the end of the protocol, all the nodes must have that piece of data in their local memory. In gossiping, all the nodes are performing a broadcast simultaneously. At the end of the protocol, all the nodes have pieces of data originated in all the other $N - 1$ nodes of the network considered.

Communication models are required in order to describe the algorithms and the time complexity of communication algorithms. The main results in the case of electronically interconnected networks can be found in [20, 39]. The results in the case of optically interconnected networks are given in Section 4.3.

Finally, two models of computation that take advantage of these communication models are introduced in Section 5. These models can be seen as extensions of the popular PRAM model.

# 2 SWITCHING TECHNIQUES

The nodes communicate with their neighbors by exchanging messages through *channels*. A channel is a one-way point-to-point connection between two nodes connected by a physical link (arc of the graph). Several channels can share the same physical link in the case of *multiplexing*.

The communication features of the interface between the memory and the communication links should also be characterized for each processor. During a communication, if each node can only send or receive one message on one link at a time, the communication is called 1-*port*. If, on the contrary, each node can simultaneously use all its links, the communications are called $\Delta$-*port*, where $\Delta$ refers to the maximum degree of the nodes in the network.

When a message is transmitted between processors that are not directly linked, the message must be routed through intermediate nodes and this routing is done with the help of routers. A router is characterized by its *switching time*, also called *latency*. Switching in a router consists of receiving a destination address, decoding the address in order to determine the appropriate output channel, and sending the message through this channel. Depending on the protocols used, switching can also include physical connection of the input link with the output link determined by the router. The latency can be only a few tens of nanoseconds in case of hardware routing but can go beyond a microsecond in the case of software routing.

## 2.1 Usual switching techniques

The various usual switching techniques are described by Kermani and Kleinrock in [38]. They are as follows.

### *Circuit-switching*

This is the principle of a telephone: a connection is established first (this means reserving a sequence of channels) and the conversation begins after.

### *Message-switching*

Messages move through the network towards their final destination by passing through intermediate nodes. At each stage, the channel used is immediately

freed. This technique is known under the name *store-and-forward* in the context of distributed machines. One flaw of this technique is the necessity of large registers for storing the message on the intermediate processors. In fact, such messages are usually stored in the global memory. However, memory access time, being proportional to the size of the messages, slow down communication dramatically.

## *Wormhole routing*

In the most recent distributed memory machines, the store-and-forward routing mode was displaced by *wormhole* routing.
Contrary to the store-and-forward mode, in which messages (or packets) are entirely stored in the memory of a processor before being transmitted to the next processor, in the wormhole routing mode the messages proceed through the processor network flit by flit (a *flit* — *flow control digit* — is the size of the buffer of a channel), with the first flit containing the destination address. The header, that is, the first flit, progress by a channel each time it is possible. The rest of the message follows, freeing the last channel which contains the end of the message. The last channel then becomes available for another message.

It is very important to distinguish this routing mode from routing by packet-switching. In the latter, each packet contains the destination address in its header and can be routed independently. In wormhole routing mode, only the first flit contains the destination address.

Now the time required to send a message of size $L$ over links having a constant bandwidth $\frac{1}{\tau}$ is considered.

In the case of store-and-forward, the transmission time of the message between two neighboring processors is given by the sum of a *start-up* (or initialization) time, denoted by $\beta$, which is the time it takes to initialize the memory registers and the time of receipt procedures, and a propagation time $L\tau$, which is directly proportional to the length $L$ of the message. Thus the cost of sending a message of size $L$ at distance $d$ is $d(\beta + L\tau)$. This cost could be decreased if one allows us to split the message in packets, and pipeline the packets along the path. For an optimal size of packets the cost is decreased to [20]: $\left(\sqrt{L\tau} + \sqrt{(d-1)\beta}\right)^2$.

In the case of wormhole or circuit-switching, one can send a message directly to a node at distance $d$ with time $\alpha + d\delta + L\tau$, where the parameter $\alpha$ is the

start-up time of the sending process and the delay $\delta$ is the time required to switch the router at each intermediate node.

In order to compare the last two models, observe that a distributed machine with circuit-switching or wormhole routing can also communicate neighbor-to-neighbor. Hence $\beta = \delta + \alpha$.

## 2.2 Wavelength Division Multiplexing

A new model of commutation is presented here, interested readers should refer to [17, 47] for an excellent introduction to the techniques. Only logical aspect of the communication are considered and we keep the optical implementations details to the minimum.

The large optical spectrum may be divided into numerous different channels, and each is assigned a different wavelength. This approach is known as WDM: Wavelength Division Multiplexing. The limit on the number of wavelengths available depends on the technology of lasers and optical filters. Technical details are out of the scope of this chapter. However, subcarrier multiplexing and electronic Time Division Multiplexing could be used within each wavelength in order to increase the number of different possible channels. This multiplexing will not slow down the communications as the interfaces of the nodes are not able to take full advantage of the bandwidth of optical fibers.

WDM lightwave networks are usually classified into *broadcast-and-select* networks and *wavelength-routing* networks. In both categories, single-hop and multihop networks [45, 46] could be considered. However, the models presented here focus on broadcast-and-select networks.

In broadcast-and-select networks, $N$ stations connected to the same network use $N$ different wavelengths to communicate via a passive network fabric (Star coupler). Each station is equipped with at least one transmitter T and one receiver R. The wavelength of each transmitter is broadcast to all receivers, see Figure 1.

The right side of the figure is an example with 3 stations and each one is built of one transmitter and one receiver. Every station emits its signal on its own wavelength, and receive all the other signals.

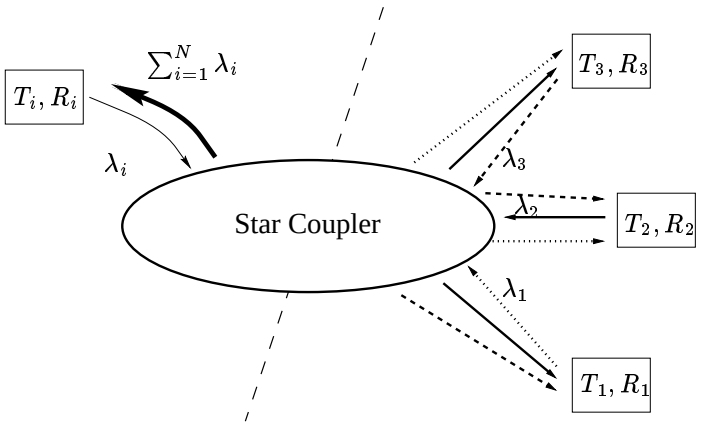



**Figure 1** Broadcast-and-select

Two types of transmitters/receivers are available: fixed or tunable wavelength types. When fixed wavelength receivers are used, the transmitters have to be tunable and should be tuned to the appropriate wavelength before each communication. In the case of tunable transmitters and fixed wavelength receivers, the source node selects the wavelength before the communication is established. The last case is when both transmitters and receivers are tunable, the arbitration protocol usually uses a control channel. Fixed wavelength devices are often chosen due to the prohibitive cost of tunable ones.

Most of the results described in the following section concern OPS based networks. Rainbow [36] is an example of a practical implementation of such network.

Usually, the number of wavelengths available to build an interconnection network is limited due to cost reasons, thus all the messages could not be delivered in one hop. Messages transit through switches and networks are said to be multi-hop. In the case of multi-hop networks, conversions from/to electronic or photonic domain are required.

Possible multi-hop network topologies are described in Section 3, and corresponding collective communication issues are given in Section 4.

The case of single-hop networks is presented in Section 4.3. These networks are also known as *all-optical* networks as messages reach their destination in one hop without being converted to electronical representation in between. When the

number of wavelengths available is not sufficient to complete the data exchange in one step, the number of additional steps required has to be minimized.

Others limitations could come from the optical power budget, indeed a minimum power is required at each receiver (dividing the signal may introduce loss). The transmitter should have a higher power.

The communication algorithm will have to take into account these different switching techniques.

# 3 TOPOLOGIES: FROM GRAPH TO HYPERGRAPH MODELS

A graph representation of interconnection networks is considered and new results in the design of topologies motivated by optical devices are given. Indeed, the way nodes are interconnected in a network is driven by technical constraints: complete interconnection is limited to a small number of nodes as each I/O port grows up the complexity of a node. Even optics has limits on the fan-out of switches.

The following definitions will be used in this chapter.

## 3.1 Definitions

The usual notations are taken from [6].

- A *directed graph* (or simply a *digraph*) $G = (V, A)$ where $V$ is called the vertex set and $A$, a multiset whose elements are from $V \times V$, is called the arc set. A *symmetric digraph* is a digraph such that if $(u, v) \in A(G)$ then $(v, u) \in A(G)$.
- The number of vertices of the graph is called its *order* and is denoted by $N$.
- $y$ is said to be a *successor* of $x$ if there is an arc $(x, y)$. The set of successors of a vertex $x$ is denoted by $\Gamma_G^+(x)$ and its cardinality, denoted by $d^+(x)$, is called the *outdegree* of $x$. The set $\Gamma_G^-(x)$ of *predecessors* of $x$ and the *indegree* $d^-(x)$ are defined similarly.

- In many cases the distinction between initial and end vertices is irrelevant. Thus the notion of an *undirected graph* is introduced: an arc $(x, y)$ is replaced by the set consisting of the two vertices $x$ and $y$, called an *edge* of the graph and denoted by $[x, y]$.

- Two vertices are *adjacent* or *neighbors* if there exists an arc or edge between them.

- Given a vertex $x$ of a graph $G$, the number of edges incident with $x$ is called the *degree* of $x$, denoted by $d_G(x)$ (or by $d(x)$ if confusion is unlikely).

  The maximum over the degrees of all vertices of $G$ is called the *maximum degree* and is denoted by $\Delta(G)$, or simply $\Delta$.

  The minimum over the degrees of all vertices of $G$ is called the *minimum degree* and is denoted by $\delta(G)$ or, simply, $\delta$.

- A *path* between two vertices $x$ and $y$ (and denoted $P(x, y)$) of a graph $G$ is a sequence $x_1, x_2, \ldots, x_k$ of vertices such that pairs of consecutive vertices are adjacent while $x_1 = x$ and $x_k = y$. A *dipath* from node $x$ to node $y$ is a directed path which consists of a set of consecutive arcs beginning in $x$ and ending in $y$.

  A path using each vertex at most once is called *elementary*. In the following all the paths considered are elementary and elementary will not be mentioned.

  The *length* of a path (resp. dipath) is the number of edges (resp. arcs) in it.

- Given two vertices $x$ and $y$ of a graph $G$, the *distance* between $x$ and $y$ is the length of a shortest path between them and is denoted by $\delta(x, y)$.

- The *diameter* of a graph $G$, denoted by $D(G)$ or, simply, $D$, if the context is clear, is the maximum of the distances $\delta(x, y)$ over all pairs of vertices of $G$.

- A *cycle* in a graph $G$ is a path whose initial and end vertices are identified. A cycle is usually meant an *elementary* cycle, that is, one using no vertex more than once.

- The *Cartesian sum*, often called *Cartesian product* or *box product*, denoted by $G \Box G'$, of two graphs $G = (V, E)$ and $G' = (V', E')$, is the graph whose vertices are the pairs $(x, x')$ where $x$ is a vertex of $G$ and $x'$ is a vertex of $G'$. Two vertices $(x, x')$ and $(y, y')$ of $G \Box G'$ are adjacent if and only if either $x = y$ and $[x', y']$ is an edge of $G'$, or $x' = y'$ and $[x, y]$ is an edge of $G$.

## 3.2 Degree vs Diameter

New optical devices, such as Optical Passive Stars (see Figure 1), bring new interests in network topologies research. Signals are broadcast simultaneously on different wavelengths and these devices could implement what is called a bus network. A bus is a multiple access medium shared among two or more nodes, whether it is based in electronics or optics (see [50] for a description of possible implementations). These networks are modeled by hypergraphs where vertices represent the processors and edges represent the buses. The following construction methods of bus networks that connect a large number of processors with a given maximum processor degree $\Delta$, a maximum bus size $r$, and a network diameter $D$ are taken from [12]. Hypergraphs are used to represent the underlying topology of the bus interconnection networks.

### $(\Delta, D, r)$-*hypergraph problem*

An *(undirected) hypergraph* $H$ is a pair $H = (\mathcal{V}(H), \mathcal{E}(H))$ where $\mathcal{V}(H)$ is a non-empty set of elements, called *vertices*, and $\mathcal{E}(H)$ is a finite set of subsets of $\mathcal{V}(H)$ called *edges*. The number of vertices in the hypergraph is $n(H) = |(\mathcal{V}(H)|$ and the number of edges is $m(H) = |\mathcal{E}(H)|$ where the vertical bars denote the cardinality of the set. The *degree* of a vertex $v$ is the number of edges containing it and is denoted by $\Delta_H(v)$. The *maximum degree* over all of the vertices in $H$ is denoted by $\Delta(H)$. The *size* of an edge $E \in \mathcal{E}(H)$ is its cardinality, and is denoted by $|E|$. The *rank* of $H$ is the size of its largest edge, and is denoted by $r(H)$. A *path* in $H$ from vertex $u$ to vertex $v$ is an alternating sequence of vertices and edges $u = v_0, E_1, v_1, \cdots, E_k, v_k = v$ such that $\{v_{i-1}, v_i\} \subseteq E_i$ for all $1 \leq i \leq k$. The *length* of a path is the number of edges in it. The *distance* between two vertices $u$ and $v$ is the length of a shortest path between them. The *diameter* of $H$ is the maximum of the distances over all pairs of vertices, and is denoted by $D(H)$.

An hypergraph with maximum degree $\Delta$, diameter $D$, and rank $r$, is called a $(\Delta, D, r)$-*hypergraph*. An example of a $(2, 2, 5)$-*hypergraph* is given in Figure 2. The problem on bus networks considered in the introduction is known as the $(\Delta, D, r)$-*hypergraph problem* and consists of finding $(\Delta, D, r)$-hypergraphs with the maximum number of vertices or finding large $(\Delta, D, r)$-hypergraphs. The maximum number of vertices in any $(\Delta, D, r)$-hypergraph is denoted by $n(\Delta, D, r)$.

In the case $r = 2$ (graph case), this problem has been extensively studied and is known as the $(\Delta, D)$-graph problem (see for example [10], [11]). The maximum

number of vertices in any $(\Delta, D)$-graph is denoted by $n(\Delta, D)$. See the table [1] and construction of such graphs detailed in [10].

Finally, let us mention that the drawing of hypergraphs can be very complex and therefore it is useful to represent an hypergraph $H$ with a bipartite graph,

$$R(H) = (\mathcal{V}_1(R) \cup \mathcal{V}_2(R), \mathcal{E}(R))$$

called the *bipartite representation graph*. Every vertex $v_i$ in $\mathcal{V}(H)$ is represented by a vertex $v_i$ in $\mathcal{V}_1(R)$ and every edge $E_j$ in $\mathcal{E}(H)$ is represented by a vertex $e_j$ in $\mathcal{V}_2(R)$. An edge is drawn between $v_i \in \mathcal{V}_1(R)$ and $e_j \in \mathcal{V}_2(R)$ if and only if $v_i \in E_j$ in $H$.

### *Moore bound*

A bound on the maximum number of vertices in a $(\Delta, D, r)$-hypergraph (analogous to the the classical Moore bound [34]) can be easily calculated: Each vertex belongs to at most $\Delta$ edges and each edge contains at most $r$ vertices. Thus there can be at most $\Delta(r-1)$ vertices at distance one from any vertex. More generally, the maximum number of vertices at distance $i$ from any vertex can be at most $\Delta(\Delta-1)^{i-1}(r-1)^i$. Therefore

$$n(\Delta, D, r) \leq 1 + \Delta(r-1) \sum_{i=0}^{D-1} (\Delta-1)^i (r-1)^i.$$

This bound is known as the *Moore bound for undirected hypergraphs*, and hypergraphs attaining it are known as *Moore geometries*.

For $D > 2$, Moore geometries cannot exist, with the exception of the cycles of length $2D+1$ (the case $\Delta = 2$ and $r = 2$). For a comprehensive survey on these results see [7]. Even for $D = 2$ and $r = 2$ (graph case), only four Moore graphs can exist.

## 3.3 Directed hypergraphs

Now let us mention the directed vs undirected question. Indeed, a lot of work has been done with undirected hypergraph models while actual problems deal with either directed topologies or symmetric directed ones. The problem of

[1] The table of the largest known $(\Delta, D)$-graphs is maintained by the group of graph researchers in Barcelona, at URL : `http://www_mat.upc.es/grup_de_grafs/table_g.html`

global communications will be studied under the directed assumption and a formal definition of directed hypergraphs will be given in the following. See Section 5 for examples.

In the directed bus networks a bus is divided into two subsets. One subset of nodes can use the bus only to send messages while the nodes of the other subset can only receive messages from the bus.

## *Definition*

A *directed hypergraph* $H$ is a pair $(\mathcal{V}(H), \mathcal{E}(H))$ where $\mathcal{V}(H)$ is a non-empty set of elements (called *vertices*) and $\mathcal{E}(H)$ is a set of ordered pairs of non-empty subsets of $\mathcal{V}(H)$ (called *hyperarcs*). If $E = (E^-, E^+)$ is a hyperarc in $\mathcal{E}(H)$, then the non-empty vertex sets $E^-$ and $E^+$ are called the *in-set* and the *out-set* of the hyperarc $E$, respectively. The sets $E^-$ and $E^+$ need not be disjoint. $|E^-|$ is the *in-size*, and $|E^+|$ is the *out-size* of hyperarc $E$. The *maximum in-size* and the *maximum out-size* of a directed hypergraph $H$ are, respectively,

$$s^-(H) = \max_{E \in \mathcal{E}(H)} |E^-| \quad \text{and} \quad s^+(H) = \max_{E \in \mathcal{E}(H)} |E^+|.$$

If $s^- = s^+ = 1$, a directed hypergraph is nothing more than a digraph.

Let $v$ be a vertex in $\mathcal{V}(H)$. The *in-degree* of $v$ is the number of hyperarcs that contain $v$ in their out-set, and is denoted by $d_H^-(v)$. Similarly, the *out-degree* of vertex $v$ is the number of hyperarcs that contain $v$ in their in-set, and is denoted by $d_H^+(v)$. The *maximum in-degree* and the *maximum out-degree* of $H$ are, respectively,

$$d^-(H) = \max_{v \in \mathcal{V}(H)} d_H^-(v) \quad \text{and} \quad d^+(H) = \max_{v \in \mathcal{V}(H)} d_H^+(v).$$

A *walk* in $H$ from vertex $u$ to vertex $v$ is an alternating sequence of vertices and hyperarcs $u = v_0, E_1, v_1, E_2, v_2, \cdots, E_k, v_k = v$ such that $v_{i-1} \in E_i^-$ and $v_i \in E_i^+$ for each $1 \leq i \leq k$. The *length* of a walk is equal to the number of hyperarcs on it. The *distance* and the *diameter* are defined analogously to those in the undirected case.

The incidence relations between the vertices and hyperarcs in a directed hypergraph $H$ are represented using a bipartite digraph,

$$R(H) = (\mathcal{V}_1(R) \cup \mathcal{V}_2(R), \mathcal{E}(R))$$

called the *bipartite representation digraph*. Every vertex $v_i$ in $\mathcal{V}(H)$ is represented by a vertex $v_i$ in $\mathcal{V}_1(R)$ and every hyperarc $E_j$ in $\mathcal{E}(H)$ is represented by a vertex $e_j$ in $\mathcal{V}_2(R)$. An arc is drawn from $v_i \in \mathcal{V}_1(R)$ to $e_j \in \mathcal{V}_2(R)$ if and only if $v_i \in E_j^-$ in $H$, and an arc is drawn from $e_j \in \mathcal{V}_2(R)$ to $v_i \in \mathcal{V}_1(R)$ if and only if $v_i \in E_j^+$ in $H$.

## 3.4 Practical topologies

Due to practical reasons, a good topology for an interconnection network is rarely the largest known $(\Delta, D)$-graph or $(\Delta, D, r)$-hypergraph. Indeed, the network has to be implemented with the available technology, and the design should be scalable for economic reasons.

Parallel computers use to have electronic based interconnection networks. Popular topologies are hypercubic networks such as hypercubes, meshes, tori, or more generally speaking $k$-ary-$n$-cubes [39]. They offer regularity, symmetry, high connectivity, fault tolerance, simple routing and also reconfigurability. In the case of the hypercube network, the logarithmic diameter is one attractive feature of the topology. However, a major drawback is the scalability of the networks: the node complexity (degree) increases with the total number of nodes in the network.

By the way, low (and/or constant) degree and small diameter networks are better candidates. This is the case for *de Bruijn* and *Kautz* graphs which are among the best known with respect to the $\Delta$ and $D$ parameters.

Once the problem is stated for graph models, one have to deal with hypergraph models. What kind of topology is best suited for bus networks? Different networks were recently proposed that take advantage of the optical issues. Bus networks are considered today because of the advantages of optics over electronics: high bandwidth, large fan-out and low signal crosstalk. In addition to the hypergraphs presented here, interested readers should refer to the table in [8].

### *Hypermeshes and hypercubes*

Mesh is probably the most popular topology in many areas of interconnection networks : VLSI routing, Wafer Scale Integration, arrays of processors, parallel computers, metropolitan networks. Main qualities are: scalability, sim-

ple routing (including deadlock-free routing), natural embedding of numerical structures (i.e. vector and matrices) on the topology, and also the easy layout of the network with the current planar technology.

Let first define the $n$-dimensional mesh denoted by $M(p_1, p_2, \ldots, p_n)$, as the cartesian sum (see definition 3.1) of $n$ paths on $p_i$ vertices, with $i = 1, 2, \ldots, n$ and $p_i \geq 2$.

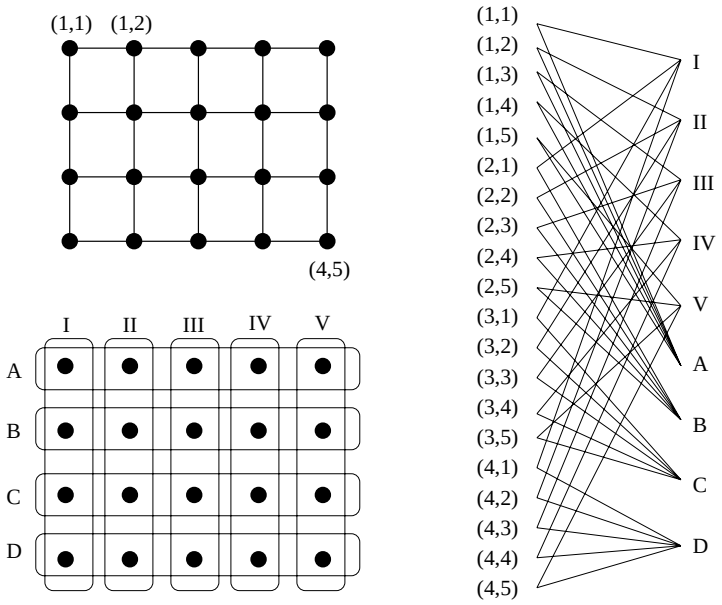



**Figure 2** Mesh $M(4,5)$, Hypermesh $HM(4,5)$ and its bipartite representation

Each vertex will naturally be denoted by a $n-tuple$ $(i_1, i_2, \ldots, i_n)$, i.e., $V(M(p_1, p_2, \ldots, p_n)) = \{(i_1, i_2, \ldots, i_n), 1 \leq i_k \leq p_k, k = 1, \ldots, n\}$.

A particular case of this graph is the hypercube, when $p_i = 2$, $i = 1, \ldots, n$. The *hypercube of dimension* $n$, denoted by $H(n)$, is a graph whose vertices are all words of length $n$ over the two-letter alphabet $\{0, 1\}$ and whose edges connect two words which differ in exactly one coordinate. Vertex $x_1 x_2 \cdots x_i \cdots x_n$ is thus joined to vertices $x_1 x_2 \cdots \overline{x_i} \cdots x_n$ with $i = 1, 2, \ldots, n$.

This graph could be seen as the cartesian sum of $n$ paths of length 2 i.e. a $n$ dimensional mesh $M(2, 2, \cdots, 2)$.

Note that people has also considered the graph based upon the cartesian sum of $n$ cycles which is called an $n$-dimensional torus.

## *Hypermeshes*

Then, the hypermeshes proposed in [50] are defined with the notation of Section 3.2 by:

- $\mathcal{V}(HM(p_1, p_2, \ldots, p_n)) = V(M(p_1, p_2, \ldots, p_n))$
- an edge contains the vertices which agree on all coordinates but one

In Figure 2 a $M(4,5)$ is represented, and two pictures of the corresponding $HM(4,5)$ are given. In the bottom picture, each set of vertices (rows and columns) is an hyperedge of $HM(4,5)$, and we label from I to V and A to D the set of hyperedges. In the bipartite representation on the left of the figure, the left column represents the set of vertices and the right column represents the set of hyperedges (crossbars). This representation is useful as it provides us with a better image of the hypergraph (degree, diameter, routing, ...).

The implementation nor the communication features of the network studied in [50] are considered here. See the chapter 9 of T. Szymanski in this book for a detailed presentation of the Hypermeshes.

## *Spanning bus hypercubes and dual bus hypercubes*

Wittíe [53] has defined two hypercube based bus networks, with generalization to $W$-wide $D$-dimensional mesh (in the actual generalized hypercube, a $W$-letter alphabet is used and the graph is defined as the iterated cartesian sum of complete graphs over $W$ vertices).

In a spanning bus hypercube all $W$ nodes aligned in the same dimension are interconnected with a bus. That is every node is connected to a different bus in each dimension. This could be a strong limiting factor for implementation, though P. Dowd [22] has described an efficient multiple access control to implement these topologies. The media access control overcomes the large degree of the graph.

In a dual bus hypercube, some buses are removed from the network in order to have only two (here dual stands for two and not duality) bus connections per node.

## *Compound techniques*

A general technique used to construct large $(\Delta, D, r)$-hypergraphs is to start from good ones for small values of $\Delta$, $D$ and $r$, then to combine them to build larger ones [12]. A first combination is the cartesian sum, but other graph products can also be used.

## *Optical Multimesh Hypercube (OMMH)*

An OMMH [41] is characterized by a triplet $(l, m, n)$ where $l$ and $m$ represents respectively the row and column dimensions of a torus, and $n$ represents the dimension of a binary hypercube. Recall a torus is similar to a mesh but defined as the cartesian sum of cycles instead of paths ($C_l \square C_m$ in the two dimensional case).

The OMMH is actually the cartesian sum of a $n$-dimensional hypercube ($H(n)$) and a 2-dimensional torus $TM(l, m)$: $H(n) \square TM(l, m)$. The aim is to find a tradeoff between the hypercube (small $\log_2 N$ diameter but large $\log_2 N$ degree) and the toroidal mesh (large $\sqrt{N}$ diameter but small constant degree).

In Figure 3 we show the example of $H(3) \square TM(2, 3)$. The representation is taken from [41]. Each dashed line represents a cycle in the torus and all the vertices crossing the line belonging to it. A. Louri and H. Sung proposed a 3-dimensional optical implementation of the network which is not a bus network, but a complex interconnection network with an efficient optical implementation. The construction could easily be extended to an hypergraph network by displacing the torus by an hypermesh of same order.

## *Partitioned Optical Passive Star (POPS)*

The interconnection network is built up with several passive star couplers. A $(n, d, r)$ POPS consists in $n$ nodes of degree $d$ linked with a redundancy factor $r$. The redundancy refers to the number of paths available between any pair of nodes. Each path traverses exactly one coupler and all couplers have equal fan-

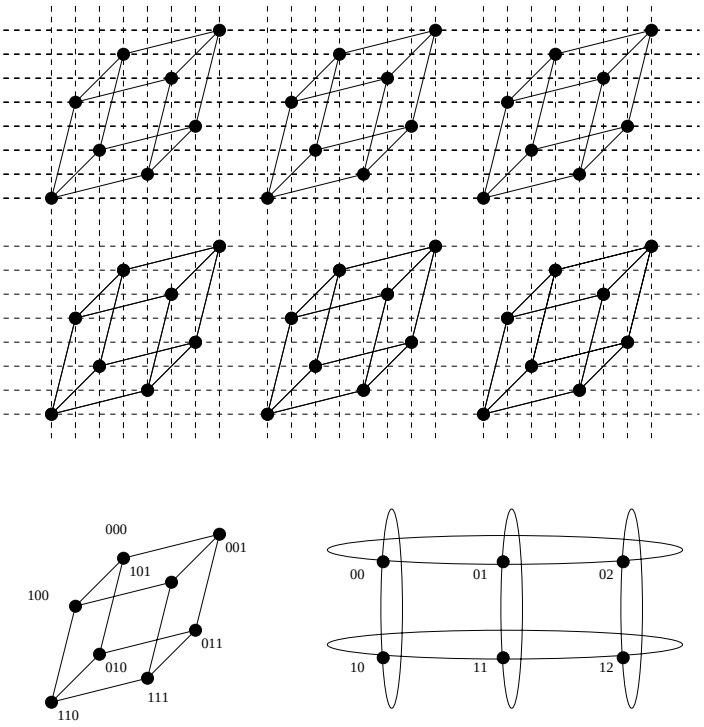



**Figure 3** Optical Multimesh Hypercube $(3, 2, 3)$

in and fan-out $d$ (in-degree and out-degree). The control and the construction of such a network is described in [18].

The $n$ nodes are partitioned in groups. We will present the case of groups of size $d$ with redundancy $r = 1$, and $d$ divides $n$. Each node is connected to every star coupler in a perfect-shuffle way. In Figure 4, the 9 nodes are partitioned in 3 groups, each group corresponding to 3 star couplers. The graph is represented as a directed bipartite graph, with a repeated set of vertices to the left and to the right of the couplers. One set represents the transmitters while the other set represents the receivers, for every node in the graph.

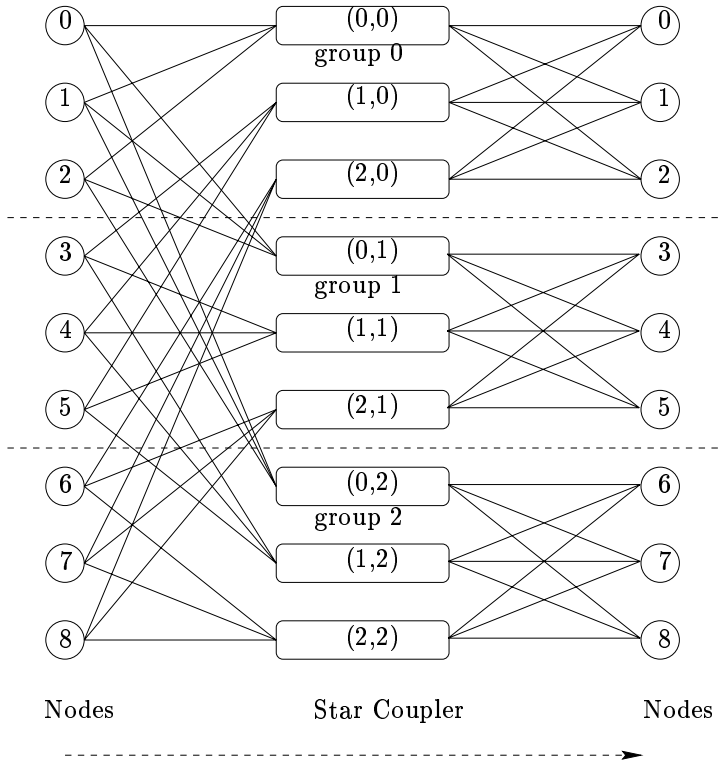



**Figure 4** Partitioned Optical Passive Star $(9, 3, 1)$

## *Stack-Graphs*

Stack-graphs [16] are obtained by piling up copies of one original graph, and by replacing each stack of edges by one hyperedge. An hyperedge contains all the extremities of the copies of one original edge.

Let $G = (V, E)$ be the original graph. The corresponding stack-graph $\varsigma(G, m)$ is defined as follows:

- $\mathcal{V}(\varsigma(G, m)) = \{0, \cdots, m-1\} \times V,\ m \geq 1$
- $\mathcal{E}(\varsigma(G, m)) = \cup_{(x,y)\in E}\mathcal{E}_{(x,y)}$ where $\mathcal{E}_{(x,y)} = \{0, \cdots, m-1\} \times \{x, y\}$

This definition allows us to derive a directed hypergraph if the original graph is a digraph.

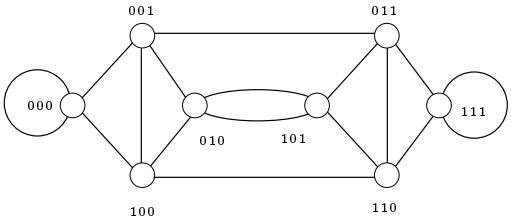
## *Dual hypergraphs*

The basic idea of the duality tool [12] is to take advantage of the properties of the best point-to-point networks to construct bus networks.

The *dual* of an hypergraph $H = (\mathcal{V}(H), \mathcal{E}(H))$ is the hypergraph $H^*$ $(= (\mathcal{V}(H^*), \mathcal{E}(H^*)))$ where the vertices of $H^*$ correspond to the edges of $H$, and the edges of $H^*$ correspond to the vertices of $H$. A vertex $e_j^*$ is a member of an edge $V_i^*$ in $H^*$ if and only if the vertex $v_i$ is a member of $E_j$ in $H$.

Consider a graph $G = (V, E)$. If we define a bus as a set of processors in which any pair of processors could communicate in one logical step, then it is natural to think to a node $v \in V$ as an hyperedge -of some hypergraph- containing all the neighbors of $v$.

We give an example of this technique and detail the two graphs in Figure 5. We take as an input graph an undirected *de Bruijn* $UB(2,3)$ and give as an output its dual hypergraph (see the bipartite representation on the right side of the figure).



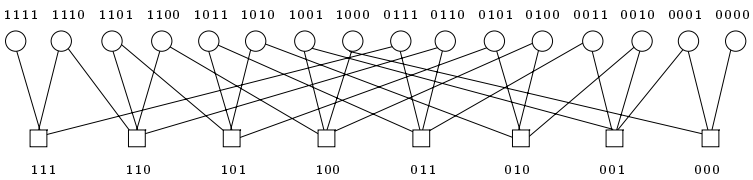




**Figure 5** Binary *de Bruijn* network $B(2,3)$ and its dual hypergraph

The *de Bruijn* digraph (resp. graph), denoted by $B(d, D)$ (resp. $UB(d, D)$), has $N = d^D$ vertices with diameter $D$ and in-degree and out-degree $d$ (resp. degree $2d$). The vertices correspond to the words of length $D$ over an alphabet of $d$ symbols. The arcs (or edges) correspond to the shift operations: Given a word $X = x_1 \cdots x_D$ on an alphabet $\mathcal{A}$ of $d$ letters, where $x_i \in \mathcal{A}$, $i = 1, 2, \cdots, D$, and given $\lambda \in \mathcal{A}$, the operation:

- $x_1 \cdots x_D \longrightarrow x_2 \cdots x_D \lambda$ is called a left shift;
- $x_1 \cdots x_D \longrightarrow \lambda x_1 \cdots x_{D-1}$ is called a right shift.

In the *de Bruijn* digraph $B(d, D)$, the successors are obtained by left-shift operations, whereas in the *de Bruijn* graph $UB(d, D)$, the neighbors are obtained by either left or right shift operations. An example of a *de Bruijn* digraph is given in Figure 5. The corresponding undirected *de Bruijn* hypergraph is obtained by transforming arcs to edges (i.e., removing the directions of the arcs). Here we do not remove the redundant edges (i.e., those with multiple occurrences in the graph, or those linking the same vertices).

Let us take the $UB(2, 3)$ of Figure 5. If we consider it as an hypergraph of rank 2, then we label each edge of $UB(2, 3)$ with the following construction. For every edge we consider the original arc which is coded on $D = 3$ digits, then we add the suffix 0 or 1 of the corresponding left shift. This set of edges gives the set of 16 vertices of the dual hypergraph $UB^*(2, 3)$. The (hyper)edges of $UB^*(2, 3)$ are made of the 8 vertices of $UB(2, 3)$ and are represented with boxes ($\Box$) in the bipartite representation of Figure 5. Note that each vertex belongs to at most 2 (hyper)edges, and that will be the case in any dual construction. The rank of the output hypergraph depends on the degree of the input graph. In the case of the *de Bruijn*, any even $r$ can be chosen.

The feasibility of such network topology with optical technology and the integration of fundamental optical operations to construct the network has been investigated by A. Louri and H. Sung [40].

These hypergraphs (and also *Kautz* hypergraphs) were proposed for virtual topologies of large optical networks by [37].

## *Bus-Mesh networks*

Tong et al [51] have proposed a network architecture which combines Time and Wavelength Division Multiplexing. This network can be represented by a directed hypergraph.

As the number of different wavelengths required for a large system is often larger than what is technically available, and as tunable transmitters are capable of tuning to only a small subset of wavelengths. The idea here is to design a multihop network.

When a node needs to transmit a message to another node which is not directly achievable (not tuned to the transmitter's wavelength), the message is relayed by intermediate nodes like in store-and-forward networks. If the number of wavelengths required by the system is still too high, Time Division Multiplexing can be used combined with each wavelength, by allocation of time slots.

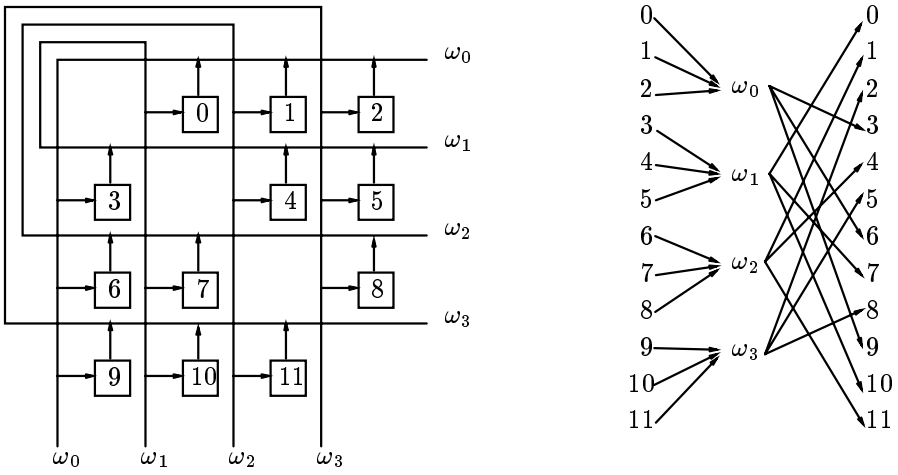



**Figure 6** Bus-Mesh network (fixed wavelength transmitter/receiver)

We will illustrate that technique on the Bus-Mesh network. The network is based on a passive star. Each node transmits and receives on two fixed different wavelengths. The time domain is divided into time slots, each slot being large

enough to contain a packet of data. We assume all the slots have the same size and are infinitely repeated in a cycle.

Figure 6 shows 12 nodes using 4 wavelengths to communicate. Each node can transmit on one wavelength and receive on a different one. The time domain is divided into 3 slots. The table of transmission cycle indicates for each time slot $t_0$, $t_1$, and $t_2$, which node is authorized to transmit on the corresponding wavelength. The arrow indicates which stations are listening on that wavelength. The bipartite directed hypergraph representation is given on the right side of the figure. For the sake of clarity, we have duplicated the vertices in two sets: transmitters and receivers.

| | $t_0$ | | | | | $t_1$ | | | | | $t_2$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\omega_0$ | 0 | $\mapsto$ | 3, | 6, | 9 | 1 | $\mapsto$ | 3, | 6, | 9 | 2 | $\mapsto$ | 3, | 6, | 9 |
| $\omega_1$ | 3 | $\mapsto$ | 0, | 7, | 10 | 4 | $\mapsto$ | 0, | 7, | 10 | 5 | $\mapsto$ | 0, | 7, | 10 |
| $\omega_2$ | 6 | $\mapsto$ | 1, | 4, | 11 | 7 | $\mapsto$ | 1, | 4, | 11 | 8 | $\mapsto$ | 1, | 4, | 11 |
| $\omega_3$ | 9 | $\mapsto$ | 2, | 5, | 8 | 10 | $\mapsto$ | 2, | 5, | 8 | 11 | $\mapsto$ | 2, | 5, | 8 |

**Table 1** Transmission cycle of a 12 nodes, 4 wavelengths Bus-Mesh network

This scheduling of time slots creates a virtual topology. Each packet includes a destination address. If node 4 wants to send a packet to node 6, it has to send the packet at time $t_1$, and only node 0 will relay the packet at time $t_0$.

The model of this network is a directed hypergraph. Indeed we can think of 4 hyperedges (associated to the 4 different wavelengths), each edge being made of an input set of vertices, and an output set of vertices.

The generalization of *de Bruijn* and *Kautz* directed hypergraphs is described in [9].

# 4 COLLECTIVE COMMUNICATIONS

We already stated the problem of collective communications in conventional point-to-point interconnection networks.

Assuming different cost functions of sending a message from an originator node to a destination node at distance $d$ from the originator, the problem is to min-

imize the total cost of the protocol (broadcasting or gossiping). This problem has been extensively studied in the case of parallel computers and interested readers can refer to [20]. Off-line problems are considered (i.e. the communications patterns are known in advance).

## 4.1 Communication models for electronic networks

### *Introduction to broadcasting with store-and-forward switching*

We assume that the protocol follows a step by step execution: during one *logical step*, every vertex can send and/or receive one elementary packet of data, according to either model $1-port$ or $\Delta-port$. The logical step ends when all the transfers are done.

In the following, the hypothesis of store-and-forward switching is used. It is easy to see that the broadcast time of an arbitrary vertex $x$ of a graph $G$ of order $N$ under the one-port full duplex constraint (denoted by $F_1$) satisfies $b_{F_1}(x) \geq \lceil \log_2 N \rceil$. Indeed, the number of vertices informed at time $t+1$ is at most twice the number of vertices informed at time $t$.

**Proposition 2** *For $N \geq 2$, $b_{F_1}(K_N) = \lceil \log_2 N \rceil$.*

PROOF. The vertices of $K_N$ are numbered from 0 to $N-1$, and we take the vertex 0 as the source of the broadcast. Consider the following broadcast protocol. At time $i \geq 1$, an informed vertex $p$ sends the message to the vertex $2^{i-1}+p$ (if $2^{i-1}+p < N$). It is easy to see, by induction, that at time $i$ all the vertices from 0 to $2^i - 1$ are informed. This guarantees broadcasting in $\lceil \log_2 N \rceil$ time units. □

### *Broadcasting in specific networks*

Unfortunately, there is no general method for computing the minimum broadcast time of a graph in this model. Each new graph is a new special case. Most of the graphs used as models of distributed architecture (meshes, tori,

hypercubes, butterfly graphs, *de Bruijn* graphs, cube-connected-cycles graphs, star-graph, and so on) have been studied and their broadcast times are known, sometimes up to a constant. The following proposition and its corollary help in finding an upper bound on broadcast time of any graph.

**Proposition 3** *In a p-ary tree of depth h, the broadcast time of the root, under the constraint $F_1$, is at most $p \times h$.*

**Corollary 2** *Let $G$ be a graph and $h$ an integer. If there is, for each vertex $x$ of $G$, a p-ary tree spanning $G$, with root $x$ and of depth $h$, then $b_{F_1}(G) \leq p \times h$.*

In particular, if we can find, for each vertex in a graph of diameter $D$, a binary spanning tree of depth $D$, then we obtain for this graph an upper bound of $2D$ on its broadcast time, that is, twice the lower bound.

## $\Delta$-*PORT*

Under the $\Delta$-port constraint (often called *shouting* and denoted $F_*$ when links are full-duplex) each processor can communicate with all its neighbors at the same time and so there is no problem in finding a broadcast algorithm for any graph. When a vertex receives a message, it sends it to all its neighbors. Thus, for a graph of diameter $D$ we have

$$b_{F_*}(G) = D.$$

## *General techniques*

A lot of work has been done on the problem of broadcasting and gossiping in point-to-point interconnection networks. However we could point out two general techniques.

In the case of store-and-forward, the general problem consists in finding the maximum number of arc-disjoint spanning trees. Then the message is cut in parts of equal sizes, and each part is "pipelined" on a different spanning tree [20].

In the case of wormhole, one could use coding theory to find optimal covering sets of the graph [21]. These sets represent the vertices that get the information at every logical step of the algorithm. The minimum number of steps is $\lceil \log_{\Delta+1} N \rceil$ when $\Delta$ concurrent communications can occur at each step.

We will see in the following that the problem is one more time different in the case of bus networks and WDM routing.

## 4.2 Collective communications in bus networks

We limit the study of collective communications in bus networks to the communication models used in the literature.

Three types of buses are considered:

- One-to-one (OTO) bus. This is the electronic model of buses. At any time two nodes belonging to the same bus could exchange data in one logical step, as if they were neighbors in a usual graph topology.

  Earlier designs of networks used this model (for instance the spanning bus hypercube or dual bus hypercube described in 2).

- One-to-all (OTA) bus, or CREW (Concurrent Read Exclusive Write). This is the motivation for bus networks: one node can broadcast its information to all the other nodes of the bus in one logical step. This has been made possible by the specificity of optical components that naturally broadcast optical signals with a large fan-out.

  Most of the models under study - such as hypermeshes - belong to that model.

- All-to-all (ATA) bus, or CRCW (Concurrent Read Concurrent Write). In this model, we suppose that all the nodes can exchange data concurrently, thus performing a "gossip" in one step. It is possible when each station connected to the bus has enough receivers (like it is the case in Lambdanet [32]), but this could be costly.

Usually, buses are 1-port: a node can communicate with only one bus at the same time even though the node could be connected to $\Delta$ buses.

The broadcasting problem is straightforward in the 1-port (OTA) model, and some work has been done on gossiping in meshes by Sotteau and Hillis [33]. Fujita and Yamashita consider the gossiping problem in mesh-bus computers (similar to squared hypermeshes: $n^2$ nodes arranged on $n \times n$ array are

connected by $n$ column-buses and $n$ row-buses. The algorithm completes the gossiping in $\lfloor n/2 \rfloor + \lceil \log_2 n \rceil + 1$ steps. A lower bound on the number of steps for this problem is shown to be $\lfloor n/2 \rfloor + \lceil \log_2 n \rceil - 1$, thus the protocol takes at most 2 more steps than an optimal algorithm.

## 4.3 Collective communications in WDM switched networks

We assess the gossiping problem in all-optical networks using wavelength division multiplexing access [17]. The problem considered is to minimize the number of wavelengths required to perform a given communication pattern between the nodes of the network using only one step (One hop problem). Indeed, under the WDM switching assumption several messages can go through a link until they do not use the same wavelength. Using the graph model, we can think of permutations with color-disjoint channels. This could be seen as a generalization of previous models in the way that wormhole in the case of WDM when the number of wavelengths (colors) equals 1, and store-and-forward in the case when the path are restricted to edges.

Routing (one-to-one communication instance of the problem) is developed in chapter 11 and the presentation here is restricted to the off-line problem, in which communications are known before we decide which wavelength has to be allocated.

Note also that we do not consider switches that use wavelength converters. When converters are available, the color of an incoming path on the switch could be changed to a different one at the output port (i.e. paths are multicolored).

In Figure 7 we give the example of five paths which have to be routed in a ring. On the left side of the figure, it is easy to check that we need 3 colors while only 2 colors are required on the right side, if we use two colors for path $(4, 1)$.

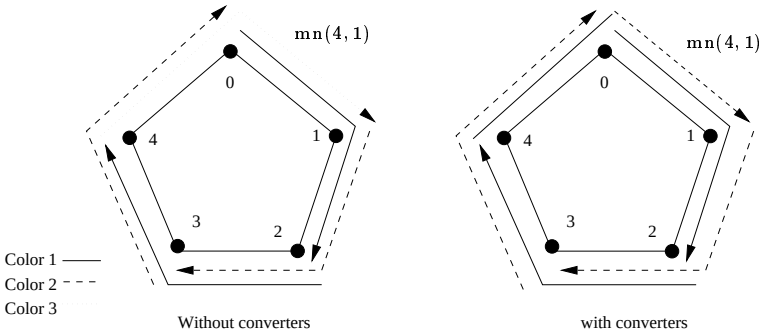



**Figure 7** Routing in a ring

## *Broadcasting and Gossiping in WDM networks*

We take the notation of [4] from which the following results are quoted. The network is modeled as a *symmetric digraph*.

## *Definition of the wavelength-routing problem*

- A *request* is an ordered pair of nodes $(x, y)$ in $G$ (corresponding to a message to be sent from $x$ to $y$).
- An *instance* $I$ is a collection of requests. Note that a given request $(x, y)$ can appear more than once in an instance.
- A *routing* $R$ for an instance $I$ in $G$ is a set of dipaths $R = \{P(x, y) \,|\, (x, y) \in I\}$.
- The *conflict graph* associated to a routing $R$ is the undirected graph $(R, E)$ with vertex set $R$ and such that two dipaths of $R$ are adjacent if and only if they share an arc of G.

In the example depicted in Figure 7, $G = C_5$ (ring of rank 5), $I = \{(i, i + 2 \mod 5), i = 0..4\}$, and $R$ is the set of paths represented with arrows in the figure.

Let $G$ be a digraph and $I$ be an instance. The *problem* $(G, I)$ asks for a routing $R$ for the instance $I$ and assigning each request $(x, y) \in I$ a wavelength, so that no two dipaths of $R$ sharing an arc have the same wavelength.

If we think of wavelengths as colors, the problem $(G, I)$ is to find a routing $R$ and a vertex coloring of the conflict graph $(R, E)$, such that two adjacent vertices are colored differently.

We denote by $\vec{\mathrm{w}}(G, I, R)$ the chromatic number of $(R, E)$, and by $\vec{\mathrm{w}}(G, I)$ (or briefly just $\vec{\mathrm{w}}$ if there is no ambiguity) the smallest $\vec{\mathrm{w}}(G, I, R)$ over all routings $R$. Thus $\vec{\mathrm{w}}(G, I, R)$ is the minimum number of wavelengths for a routing $R$ and $\vec{\mathrm{w}}(G, I)$ the minimum number of wavelengths over all routings for $(G, I)$.

Any routing by undirected paths induces a routing by directed paths, and a coloring of the undirected paths is also a coloring of the directed paths, as two edge-disjoint paths will become two arc-disjoint dipaths. Hence $\vec{\mathrm{w}}(G, I) \leq \mathrm{w}(G, I)$ for any problem $(G, I)$, and every upper bound on w is an upper bound on $\vec{\mathrm{w}}$.

## *Off-line communication problems*

As in the case of store-and-forward and wormhole models, the following special instances of the routing problem are considered:

- The *All–to–All* instance, or *gossiping* instance, denoted $I_{\mathrm{ATA}}$: $I_{\mathrm{ATA}} = \{(x, y) \mid x \in V(G),\ y \in V(G), x \neq y\}$.
- The *One–to–All* instance, or *broadcasting* instance, denoted $I_{\mathrm{OTA}}$: $I_{\mathrm{OTA}} = \{(x_0, y) \mid y \in V(G), y \neq x_0\}$, where $x_0 \in V(G)$. A *One–to–Many* instance, or *multicasting* instance is a subset of some instance $I_{\mathrm{OTA}}$.
- A *k-relation* is an instance $I_k$ in which each node is a source and a destination of no more than $k$ requests. A 1-relation is also known as a *permutation* instance. Note also that the instance $I_A$ is an $(N-1)$-relation.

## *The load parameter of a network*

- Given a network $G$ and a routing $R$ for an instance $I$, the *load* of an arc $\alpha \in A(G)$ in the routing $R$, denoted by $\vec{\pi}(G, I, R, \alpha)$, is the number of dipaths of $R$ containing $\alpha$. The *load* (also called *congestion*) of $G$ in the routing $R$, denoted by $\vec{\pi}(G, I, R)$, is the maximum load of any arc of $G$ in the routing $R$, that is, $\vec{\pi}(G, I, R) = \max_{\alpha \in A(G)} \vec{\pi}(G, I, R, \alpha)$.
- The *load* of $G$ for an instance $I$, denoted by $\vec{\pi}(G, I)$, or $\vec{\pi}$ if there is no ambiguity, is the minimum load of $G$ in any routing $R$ for $I$, that is,

$\vec{\pi}(G,I) = \min_R \vec{\pi}(G,I,R)$. For the All–to–All instance $I_{\mathrm{ATA}}$, $\vec{\pi}(G, I_{\mathrm{ATA}})$ (respectively $\pi(G, I_{\mathrm{ATA}})$) is called the *arc forwarding index* (resp. *edge forwarding index*, see [33, 49]) of $G$.

The relevance to the problem of this parameter is shown by the following lemma:

**Lemma 1** *$\vec{\mathrm{w}}(G,I) \geq \vec{\pi}(G,I)$ for any instance $I$ in any network $G$.*

In other words, to solve a given problem $(G,I)$ one has to use a number of wavelengths at least equal to the maximum number of dipaths having to share an arc.

In general, minimizing the number of wavelengths is not the same problem than realizing a routing that minimizes the number of dipaths sharing an arc (congestion). Indeed, the problem is made much harder due to the further requirement of wavelengths assignment on the dipaths. In order to get equality in Lemma 1, one should find a routing $R$ such that $\vec{\pi}(G,I,R) = \vec{\pi}(G,I)$, for which the associated conflict graph is $\vec{\pi}(G,I)$–vertex colorable.

**Question 4** *Does there always exist a routing $R$ such that the two conditions hold simultaneously: $\vec{\pi}(G,I,R) = \vec{\pi}(G,I)$ and $\vec{\mathrm{w}}(G,I,R) = \vec{\mathrm{w}}(G,I)$?*

**Theorem 5** ([5]) *Determining $\vec{\pi}(G,I)$ in the general case is NP–complete.*

**Sketch of proof.** First observe that determining $\vec{\pi}(G,I)$ is equivalent to solving the integral multicommodity directed flow problem with constant capacities. It is shown in [25] that this problem is NP–complete even for two commodities and all capacities equal to one. □

For some special cases, $\vec{\pi}(G,I)$ can be efficiently determined. This is obviously the case for trees, where routing is always unique. This is also the case of the One–to–Many instances where the problem can be reduced to a flow problem (in the graph obtained from $G$ by considering the sender node as the source, giving a capacity $\vec{\pi}$ to each arc of $G$, and joining all the vertices of $G$ to a sink $t$ with arcs of capacity 1).

The load $\pi(G,I)$ can be defined analogously for an undirected graph and it is proven that $\vec{\pi}(G,I) \leq \pi(G,I) \leq 2\vec{\pi}(G,I)$. For One–to–Many instances $I$, one can also show that $\vec{\pi}(G,I) = \pi(G,I)$.

**Question 6** *Does the equality* $\vec{\pi}(G, I_{\mathrm{ATA}}) = \lceil \pi(G, I_{I_{\mathrm{ATA}}})/2 \rceil$ *always hold ?*

## *Arbitrary network topologies*

## *Arbitrary instances*

For a general network $G$ and an arbitrary instance $I$, the problem of determining $\vec{\mathrm{w}}(G, I)$ has been proved to be NP–hard in [23]. In particular, it has been proved that determining $\vec{\mathrm{w}}(G, I)$ is NP–hard for trees and cycles. In [24] these results have been extended to binary trees and meshes. NP–completeness results in the undirected model were known much earlier (actually, well before the advent of the WDM technology). In particular, in [31] it is proved that the problem of determining $\mathrm{w}(G, I)$ is NP–complete for trees. This result has been extended in [23] to cycles, while in [24] it has been proved that the problem is efficiently solvable for bounded degree trees.

In view of this last result and of the NP–hardness of determining $\vec{\mathrm{w}}(G, I)$ for binary trees, it might seem that the problem of computing $\vec{\mathrm{w}}(G, I)$ is harder than that of computing $\mathrm{w}(G, I)$. This is not true in general. For instance, the determination of $\mathrm{w}(G, I)$ remains NP–complete when $G$ is a star network, whereas $\vec{\mathrm{w}}(G, I)$ can be efficiently computed. Indeed, in the undirected model this problem corresponds to an edge-coloring a multigraph [35], each node of which corresponds to a branch in the star network. In the directed case, the same problem becomes equivalent to an edge-coloring of a bipartite multigraph, and the problem is efficiently solvable by König's theorem.

In [1] an upper bound in the undirected model is given, the same holds also in the directed case:

**Theorem 7** *(Aggarwal et al. [1]). For any problem* $(G, I)$*, where* $G$ *has* $m$ *arcs,* $\vec{\mathrm{w}}(G, I) \leq 2\vec{\pi}(G, I)\sqrt{m}$.

Let $R$ be a routing for an instance $I$ in a network $G$. Let $L$ be the maximum length of its dipaths and $\Delta$ the maximum degree of its conflict graph. It is clear that $\Delta \leq L\vec{\pi}(G, I, R)$. By a greedy coloring we know that $(\Delta + 1)$ wavelengths are sufficient to solve the problem $(G, I)$. Thus $\vec{\mathrm{w}} = O(L\vec{\pi})$ and similarly $\mathrm{w} = O(L\pi)$. A set of critical undirected problems which reach asymptotically this upper bound (and that of Theorem 7) has been given in mesh-like networks (see [1]). By adapting their examples (orienting alternately the vertical links

up and down), the same result is obtained for the general case (not symmetric) digraphs:

**Theorem 8** *For every $\pi$ and $L$, there exists a directed graph $G$ and an instance $I$ such that $\vec{\pi}(G,I) = \pi$, $L = \max_{(x,y)\in I} \delta(x,y)$ and $\vec{\mathrm{w}}(G,I) = \Omega(\pi L)$.*

**Question 9** *Does Theorem 8 hold for symmetric digraphs?*

## *Specific instances*

The following theorem gives the exact value of $\vec{\mathrm{w}}(G, I_{\mathrm{OTA}})$ for a worst case instance $I_{\mathrm{OTA}}$ in various classes of important networks, namely the *maximally arc connected* digraphs, including the wide class of vertex transitive digraphs. A digraph $G$ is maximally arc connected if its minimum degree is equal to its arc connectivity.

**Theorem 10** *(Bermond et al. [13]). For a worst case One–to–All instance $I_{\mathrm{OTA}}$ in a maximally arc connected digraph $G$ of minimum degree $d(G)$,*

$$\vec{\mathrm{w}}(G, I_{\mathrm{OTA}}) = \vec{\pi}(G, I_{\mathrm{OTA}}) = \left\lceil \frac{N-1}{d(G)} \right\rceil \ .$$

In addition, an efficient network flow based algorithm is given to solve the problem $(G, I)$ with $\vec{\mathrm{w}}(G, I)$ wavelengths, for any One–to–Many instance $I$ in any network $G$.

**Theorem 11** *(Beauquier et al. [5]). $\vec{\mathrm{w}}(G, I) = \vec{\pi}(G, I)$, for any One–to–Many instance $I$ in any digraph $G$.*

Many other instances are relevant of interest, however a lot of practical communication problems have to deal with the on-line hypothesis, and efficient algorithms have to be designed. The techniques used are different and are not presented in this chapter.

# 5 INTRODUCING OPTICAL COMMUNICATIONS IN GENERAL MODELS OF PARALLELISM

In this section, we review the different models of parallelism involved by using optical technologies. These models are mainly derived from the PRAM model. In the following, we recall what is the PRAM model and its optical extensions, namely the OCPC model in Section 5.2 and the OPS model in Section 5.3.

## 5.1 The Parallel Random Access Machine

The PRAM model (Parallel Random Access Machine) was one of the first model for designing parallel algorithms. It presents the parallel extension of the sequential RAM model [2] in order to design parallel algorithms. The **PRAM** model consists of $P$ processors and $M$ memory modules. In a unit time step, all the processors can access the data in one of the memory modules (Read operation), perform a simple computation on their local registers, and store the result in one of the memory module (Write operation). Problems arise when several processors want to access the same memory location. In 1982, Snir proposed a classification of the PRAM models in terms of the possibility of multiple reads and writes [48], namely, *Exculsive*, when only one processor can access to a Read or Write ressource at the same time, and *Concurrent* in the other case. In case of concurrent multiple write, the result has to be clearly specified [27]. Let note, for example, the COLLISION rule, where a special character is put in a memory cell after a write conflict.

The PRAM model is indeed a theoretical model for a parallel machine. However, it does not take into account the physical network the parallel machine uses to communicate. Several extensions such as the XRAM model solve partially the problem [19].

## 5.2 The Optically Connected Parallel Computer

In 1988, Anderson and Miller proposed another way of resolving the Write conflicts, based on optical assumptions. The general model is as follows. The **Local Memory PRAM** [3] consists of a collection of processors and a col-

lection of memory modules. Each processor and each memory module has one optical receptor and one optical transmitter. Each transmitter can be focused on a receptor in one unit time. Two light beams do not interfere unless they are focused on the same receptor.

Based on these assumptions, Valiant proposed the **seclusive-PRAM** [52] (S*-PRAM). This model is now called the **completely connected Optical Communication Parallel Computer**. Using this model, it is very easy to implement an acknowledgment process. All the processors that have received a message without error can send back an acknowledgment message to the originator. This latter message will reach its destination without any conflict since processors send messages to a single destination. Thus, in a unit time, it can be assumed that all the processors know whether they have succeeded in accessing a memory module. The acknowledgment can also include the data that the processor was asking for.

The power of this model has been widely discussed [3, 26, 28–30, 52] and we will review in the following some of the principal results of this model.

## *Relations with the PRAM models*

The first problem when defining a new model is to compare it to the others and to know where it stands in the model hierarchy. In this extent, MacKenzie and Ramachandran [43] show that the OCPC is equivalent to the Exclusive Read, Concurrent Write PRAM model, whenever the contention resolutions are solved in the same way. This latter model has not been considered by Snir in his classification [48], since the Read operation in a memory cell is usually less constrained than the Write operations.

Valiant [52] described a simulation of an EREW PRAM on an OCPC. He gave a constant delay simulation of BSP (Bulk Synchronous Parallel) computer on the OCPC, connected to a randomized simulation of a $n \log n$ processor EREW PRAM on a an $n$ processor BSP. A direct simulation was given by Geréb-Grauss and Tsantilas [26] with delay $O(\log n \log\log n)$. Different works have provided such a simulation in expected delay $\Theta(\log\log n)$, but they require $n^{\Omega(1)}$ storage at each processor [42, 44]. Goldberg, Matias and Rao found a randomized simulation of an $n \log\log n$ processor EREW PRAM on an $n$ processor OCPC in $O(\log\log n)$ expected delay [30].

### *The h-relation problem*

In the $h$-relation problem, each processor has at most $h$ messages to send and $h$ messages to receive. A 1-relation is indeed a (partial) permutation, and can be easily realized in one step on the OCPC model. When the communication pattern is known *a priori*, the $h$ relation can be decomposed into $h$ partial permutation and thus can be performed within $h$ steps.

When studying the *on-line* problem, and probabilistic algorithms have to be used in order to derive good bounds. In [29], Goldberg, Jerrum and MacKenzie show that the expected number of communication steps required to route an arbitrary $h$ relation is $\Omega(h + \sqrt{\log\log n})$ on a $n$-processor OCPC. In [28], Goldberg et al. solved this problem in time $O(h + \log\log n)$.

## 5.3 The Optical Passive Star

As shown previously, the OCPC model does not take into account all the capabilities of the optical devices. Especially, this model is always a point-to-point model, i.e., in a single step, a processor can send a message to only one receiver. Using Optical Passive Stars, this constraint can be eliminated by defining the OPS model [14]. Some variations on this model have been given, e.g., see [15].

Each processor in a OPS-based network consists of a receiver, a transmitter, and a processing element with a memory unit. The transmitters and receivers may tune to any one of a range of wavelengths. All the receivers and transmitters are connected to an optical coupler, which is an all-to-all communication device with broadcast capability. Thus broadcast becomes an elementary operation, whereas in the point-to-point models such as the OCPC, this is an elaborated operation. If several messages are transmitted simultaneously on the same wavelength, then detectable "noise" is received. The abstract OPS model comprises a finite number of processors and a finite number of wavelengths.

Readers are referred again to [45] for a comprehensive survey on the implementation of such a model. In the remainder of the section, we briefly present some results on this model. Details can be found in [14].

In the following, we consider the OPS as a variant of the PRAM model with a global memory of linear size. A OPS is said to be balanced if it has as many available wavelengths as processors. Scaling problems towards realistic models

is the point of Section 5.3, since we have seen in the previous sections that the number of wavelengths is the critical resource of such a system.

## *The OPS in the PRAM hierarchy*

This new abstraction of a PRAM, dealing with new means of communication can be compared to usual models of CRCW PRAM. More precisely, it can be stated the equivalence with the COLLISION CRCW PRAM. This shows that the OPS is completely different from the OCPC model.

**Theorem 12** ([14]) *For integers $n$ and $m$ with $1 \leq m \leq n$, the $n$ processor OPS with $m$ wavelengths and the $n$ processor COLLISION PRAM with $m$ global memory cells are equivalent, i.e., each machine can simulate the other one in a step-by-step fashion with a constant slowdown. In particular, the balanced $n$ processor OPS an the balanced $n$ processor COLLISION PRAM are equivalent.*

## *Self simulation of the OPS model*

The self-simulation of a parallel model is directly related to the efficiency and ease of algorithm design: it is desirable that the algorithm designer may assume that as many processors as required by his algorithm are simultaneously available for his program. Once the algorithm designed for $kn$ processors is executed on a given machine with only $n$ processors, a simulation of the $kn$ processor program should be done by the actual $n$ processors. Moreover, considering the OPS model, the set of wavelengths should also be reduced, since the wavelengths are considered as a scarce resource. Similarly, the communication carried out by the larger virtual machine has to be scaled down to the smaller real machine.

The self-simulation is a step-by-step simulation, so that each (physical) processor simulates the operations of $k$ fixed simulated (virtual) processors. Thus the simulation of the tuning and computation phases of each step can be trivially done in $k$ steps of the simulating machine. The main problem that is addressed in the following is to simulate the communication steps.

The main results in [14] show that the OPS does exhibit scalability properties, i.e., a balanced $kn$ processor OPS can be simulated by a balanced $n$ processor OPS in a step-by-step manner with a slowdown of $O(k + \log^* n)$ with high probability. The algorithm consists essentially of a randomized solution to a

distributed load balancing problem, added to a deterministic self-simulation algorithm that takes $O(k^2)$ steps in the worst case. From this randomized solution, we can derive a deterministic off-line algorithm which is able to complete the self-simulation problem in $O(k)$ steps.

Note that these solutions use some redirection of the messages in order to balance the requests. If we consider only direct self-simulation, i.e., a message must reach its final destination with no intermediate stop-overs, then the slowdown is more important, even in the off-line case. In this particular case, $\Omega(\min\{k^2, k + \log n / \log\log n\})$ is a lower bound, even in the off-line case, or when the number of wavelengths in unbounded. A simple algorithm with slowdown $O(k^3)$ can be established by scheduling all the possible combinations independently.

# 6 CONCLUSION

Since the speed of parallel algorithms is mainly constrained by interprocessor communications, models for communications must be established first in order to provide abstract models of computation for parallel architectures. These models are dictated by the underlying network topology and a great deal of work has been undertaken in this area, motivated by the existence of parallel computers.

Three aspects of interconnection networks models have been introduced in this chapter: topologies, communications and computations. Our goal was to state what changes in optical communication systems with respect to electronical ones, and we were able to provide some answers.

Concerning topologies, a hypergraph model was introduced, that gives a better representation of the network than the usual graph model. The communication models presented rely on the use of the wavelength division multiplexing switching technique. The important parameter in all the communication problems under study is the number of different wavelengths required to perform the communication. It is also the case for the computational models derived from these communication models.

## Acknowledgements

We wish to thank Bruno Beauquier and Jean-Claude Bermond for helpful discussions and contributions. We acknowledge also the French groups ROI and RUMEUR.

## REFERENCES

[1] A. Aggarwal, A. Bar-Noy, D. Coppersmith, R. Ramaswami, B. Schieber, and M. Sudan. Efficient routing in optical networks. *Journal of the ACM*, 46(6):973–1001, November 1996.

[2] A. Aho, J. Hopcroft, and J. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley Publishing Co., 1974.

[3] R. Anderson and G. Miller. Optical communication for pointer based algorithms. Technical Report CRI 88-14, Computer Science Department, University of Southern California, Los Angeles, CA 90089-0782 USA, 1988.

[4] B. Beauquier, J.-C. Bermond, L. Gargano, P. Hell, S. Pérennes, and U. Vaccaro. Graph problems arising from wavelength–routing in all–optical networks. In *Proc. Conference WOCS97, Geneva*, April 1997.

[5] B. Beauquier, P. Hell, and S. Pérennes. Optimal wavelength-routed multicasting. Discrete Applied Mathematics, to appear.

[6] C. Berge. *Graphs and Hypergraphs*. North-Holland, 1973.

[7] J.-C. Bermond, J. Bond, M. Paoli, and C. Peyrat. Graphs and interconnection networks: diameter and vulnerability. In E. Lloyd, editor, *Surveys in Combinatorics, Invited Papers for the Ninth British Combinatorial Conference*, volume 82 of *London Math. Society Lecture Note Series*, pages 1–30. Cambridge University Press, 1983.

[8] J.-C. Bermond, J. Bond, and C. Peyrat. Interconnection network with each node on two buses. In *Proc. of the Internat. Workshop on Parallel Algorithms & Architectures, Luminy France.*, pages 155–167. North Holland, April 1986.

[9] J.-C. Bermond, R. Dawes, and F. Ergincan. de Bruijn and Kautz bus networks. Technical Report 94-32, I3S (to appear in Networks), 1994.

[10] J.-C. Bermond, C. Delorme, and J.-J. Quisquater. Strategies for interconnection networks: Some methods from graph theory. *Journal of Parallel and Distributed Computing*, 3:433–449, 1986.

[11] J.-C. Bermond, C. Delorme, and J.-J. Quisquater. Table of large $(\Delta, d)$-graphs. *Discrete Applied Mathematics*, 37/38:575–577, 1992.

[12] J.-C. Bermond and F. Ergincan. Bus interconnection networks. *Discrete Applied Mathematics*, (68):1–15, 1996.

[13] J.-C. Bermond, L. Gargano, S. Perennes, A. A. Rescigno, and U. Vaccaro. Efficient collective communication in optical networks. *Lecture Notes in Computer Science*, 1099:574–585, 1996.

[14] P. Berthomé, T. Duboux, T. Hagerup, I. Newman, and A. Schuster. Self-simulation for the passive optical star model. In P. Spirakis, editor, *European Symposium on Algorithms*, number 979 in Lecture Notes in Computer Science. Springer-Verlag, 1995.

[15] P. Berthomé and A. Ferreira. Communication issues in parallel systems with optical interconnections. *International Journal of Foundations of Computer Science*, 1997. To appear in Special Issue on Interconnection Networks.

[16] H. Bourdin, A. Ferreira, and K. Marcus. A comparative study of one-to-many WDM lightwave interconnection network for multiprocessors. In *Second International Workshop on Massively Parallel Processing using Optical Interconnections*, pages 257–264, San Antonio (USA), October 1995. IEEE Press.

[17] C. A. Brackett. Foreword. is there an emerging concensus on WDM networking. *Journal of Lightwave Technology*, 14(6):936–941, June 1996.

[18] D. M. Chiarulli, S. P. Levitan, R. P. Melhem, J. P. Teza, and G. Gravenstreter. Partitioned Optical Passive Stars (POPS) multiprocessor interconnection networks with distributed control. *Journal of Lightwave Technology*, 14(7):1601–1612, July 1996.

[19] M. Cosnard and A. Ferreira. Designing parallel non numerical algorithms. In G. J. D.J. Evans and H. Liddell, editors, *Parallel Computing'91*, pages 3–18. Elsevier Science Publishers B.V., 1992.

[20] J. de RUMEUR. *Communications dans les réseaux de processeurs.* Masson, Paris, 1994.

[21] O. Delmas and S. Perennes. Circuit-Switched Gossiping in 3-Dimensional Torus Networks. In L. Bougé, P. Fraigniaud, A. Mignotte, and Y. Robert, editors, *Proceedings of the Euro-Par'96 Parallel Processing / Second International EURO-PAR Conference*, volume 1123 of *Lecture Notes in Computer Science*, pages 370–373, Lyon, France, Aug. 1996. Springer Verlag.

[22] P. W. Dowd. Wavelength division multiple access channel hypercube processor interconnection. *IEEE Transactions on Computers*, 41(10):1223–1241, October 1992.

[23] T. Erlebach and K. Jansen. Scheduling of virtual connections in fast networks. In *Proc. of Parallel Systems and Algorithms (PASA)*, pages 13–32, 1996.

[24] T. Erlebach and K. Jansen. Call scheduling in trees, rings and meshes. In *Proc. of HICSS*, 1997.

[25] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM J. of Computing*, 5(4):691–703, Dec. 1976.

[26] M. Geréb-Grauss and T. Tsantilas. Efficient optical communication in parallel computers. In *ACM Symposium on Parallel Algorithms and Architectures*, pages 41–48, June 1992.

[27] J. Gil and Y. Matias. Fast and efficient simulations among CRCW PRAMs. *Journal of Parallel and Distributed Computing*, 23(2):135–148, Nov. 1994.

[28] L. Goldberg, M. Jerrum, T. Leighton, and S. Rao. A doubly logarithmic communication algorithm for the completely connected optical communication parallel computer. In *ACM Symposium on Parallel Algorithms and Architectures*, pages 300–309, June 1993.

[29] L. Goldberg, M. Jerrum, and P. MacKenzie. An $\Omega(\sqrt{\log \log n})$ lower bound for routing in optical networks. In *ACM Symposium on Parallel Algorithms and Architectures*, June 1994.

[30] L. Goldberg, Y. Matias, and S. Rao. An optical simulation of shared memory. In *ACM Symposium on Parallel Algorithms and Architectures*, June 1994.

[31] M. C. Golumbic and R. E. Jamison. The edge intersection graphs of paths in a tree. *J. of Combinatorial Theory, Series B*, 38:8–22, 1985.

[32] M. S. Goodman, H. Kobrinski, M. P. Vecchi, R. M. Bulley, and J. L. Gimlett. The lambdanet multiwavelength network: Architecture, applications, and demonstrations. *IEEE Journal on Selected Areas in Communications*, 8(6):995–1004, August 1990.

[33] A. Hily and D. Sotteau. Gossiping in d-dimensional mesh-bus networks. *Parallel Processing Letter*, 6(1):101–113, March 1996.

[34] A. Hoffman and R. Singleton. On Moore graphs with diameters 2 and 3. *IBM J. Research and Development*, 4:497–504, 1960.

[35] I. Holyer. The NP–completeness of edge coloring. *SIAM J. of Computing*, 10(4):718–720, 1981.

[36] F. J. Janniello, R. Ramaswami, and D. G. Steinberg. A prototype circuit-switched multi-wavelength optical metropolitan-area network. *Journal of Lightwave Technology*, May/June 1993.

[37] S. Jiang, T. E. Stern, and E. Bouillet. Design of multicast multilayered lightwave networks. IEEE GLOBECOM'93 Houston Texas, pages 452-457, 1993.

[38] P. Kermani and L. Kleinrock. Virtual cut-through: a new computer communication switching technique. *Computers Networks*, 3:267–286, 1979.

[39] F. T. Leighton. *Introduction to parallel algorithms and architectures.* Morgan Kaufmann, 1992.

[40] A. Louri and H. Sung. Optical binary de bruijn networks for massively parallel computing: Design methodology and feasibility study. Technical Report AZ85721, University of Arizona, 1994.

[41] A. Louri and H. Sung. Scalable optical hypercube-based interconnection network for massively parallel computing. *Applied Optics*, 33(32):7588–7598, November 1994.

[42] P. MacKenzie, C. Plaxton, and R. Rajamaran. On contention resolution protocols and associated probabilistic phenomena. In *ACM Symposium On Theory of Computing*, 1994.

[43] P. D. MacKenzie and V. Ramachandran. ERCW PRAMs and optical communications. In *EUROPAR: Parallel Processing, 2nd International EURO-PAR Conference.* LNCS, 1996.

[44] F. Meyer auf der Heide and C. S. Scheiderler. Fast simple dictionaries and shared memory simulation on distributed memory machines; upper and lower bounds. Pre-print, 1994.

[45] B. Mukherjee. WDM-based local lightwave networks, Part I: Single-hop systems. *IEEE Network Magazine*, 6(3):12–27, May 1992.

[46] B. Mukherjee. WDM-based local lightwave networks, Part II: Multi-hop systems. *IEEE Network Magazine*, 6(4):20–32, July 1992.

[47] R. Ramaswami. Multiwavelength lightwave networks for computer communication. *IEEE Communications Magazine*, pages 78–88, February 1993.

[48] M. Snir. On parallel searching. *SIAM Journal of Computing*, 14(4):688–708, Aug. 1985. Also appeared in ACM Symposium on Principles of Distributed Computing,1982.

[49] P. Solé. Expanding and forwarding. *Discrete Applied Mathematics*, 58:67–78, 1995.

[50] T. Szymanski. "Hypermeshes": Optical interconnection networks for parallel computing. *Journal of Parallel and Distributed Computing*, 26(1):1–23, April 1995.

[51] S.-R. Tong, D. H. C. Du, and R. J. Vetter. Design principles for multi-hop wavelength and time division multiplexed optical passive star networks. *IEEE Journal on Selected Areas in Communications*, 4(2), 1995.

[52] L. Valiant. General purpose parallel architectures. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, pages 943–971. Elsevier/MIT Press, 1990.

[53] L. D. Wittíe. Communication structures for large networks of microcomputers. *IEEE Transactions on Computers*, C-30(4):264–273, April 1981.