Parallel Processing Letters, Vol. 12, No. 1 (2002) 41–50

ANALYSIS OF A LATENCY HIDING BROADCASTING ALGORITHM ON A RECONFIGURABLE OPTICAL INTERCONNECT

AHMAD AFSAHI
Department of Electrical and Computer Engineering, Queen's University
Kingston, Canada K7L 3N6
ahmad@ee.queensu.ca

NIKITAS J. DIMOPOULOS
Department of Electrical and Computer Engineering, University of Victoria
P.O. Box 3055, Victoria, Canada V8W 3P6
nikitas@ece.uvic.ca

Received September 2000
Revised March 2002
Accepted by S. Akl

ABSTRACT

Free-space optical interconnection is used to fashion a reconfigurable network. Since network reconfiguration is expensive compared to message transmission in such networks, latency hiding techniques can be used to increase the performance of collective communications operations. Berthome and Ferreira have recently proposed a broadcasting algorithm for their loosely-coupled optically reconfigurable parallel computer where they have shown that the total number of nodes, *N(S)*, informed up to step *S* follows a recurrence relation. We have adapted their algorithm to our reconfigurable optical network, *RON (K, N)*, which has a slightly different modeling. We present a new analysis of this broadcasting algorithm on our network. This paper contributes by providing closed formulations for the *N(S)* that yield the termination time for both single-port and *k*-port modeling. The derived closed formulae are easier to compute than the recurrence relations.

Keywords: Broadcasting; collective communications; latency hiding; reconfigurable optical interconnect.

1 Introduction

Message-passing multicomputers are composed of a number of computing modules that communicate with each other by exchanging messages through their interconnection networks. As the communication overhead is one of the most important factors affecting the performance of parallel computers [1], there has been a growing interest in the design of interconnection networks [2].

A complete interconnection network is desirable where any computing node can communicate with any other node in a single-hop. However, implementing complete networks using metal-based interconnections is not feasible when the number of computing nodes is very large as there is a fixed physical link between any two nodes.

Optics is ideally suited for implementing interconnection networks because of its superior characteristics over electronics [3, 4]. Free-space reconfigurable optical interconnects have the potential to solve the problems associated with implementing complete networks due to their ability to reconfigure the interconnect. Currently, several research groups in academia and industry are working on different aspects of utilizing optical interconnects in parallel processing systems [5, 6].

Collective communications are common basic patterns of interprocessor communication that are frequently used as building blocks in a variety of parallel algorithms [7]. For instance, in *broadcasting*, a node sends its unique message to all other nodes. Broadcasting is used in a variety of linear algebra algorithms, database queries and transitive closure algorithms. Proper implementation of these communication operations is one of the key to the efficient execution of parallel algorithms. The growing interest in collective communications is evident by their inclusion in the Message Passing Interface (MPI) [8].

Numerous works have been reported on collective communications. Excellent surveys on collective communication algorithms can be found in [9, 10]. In the context of optical interconnection networks, Berthome and Ferreira have presented broadcasting and multicasting algorithms for networks using *optical passive stars* (OPS) [11, 12]. Gravenstreter and Melhem have presented some communication algorithms in *partitioned optical passive stars* (POPS) networks [13].

The authors in [11] have proposed a broadcasting algorithm for their *loosely-coupled optically reconfigurable parallel computer*, *ORPC (k)*, where they have shown that the total number of nodes, *N(S)*, informed up to step *S* follows a recurrence relation. We have adapted their algorithm to our *reconfigurable optical network*, *RON (K, N)*, which has a slightly different modeling where only the sender is allowed to reconfigure and hence the delay penalties occur there. However, the receiver is entirely passive. The main contribution of this paper is the analysis of this broadcasting algorithm on our network where it provides closed formulations for *N(S)* that yield the termination time for both single-port and *k*-port modeling. The derived closed formulae are easier to compute than the recurrence relations.

In Section 2 of this paper, we introduce a complete and reconfigurable optical interconnection network. We discuss its communication modeling in Section 3. In Section 4, we present the analysis of the broadcasting algorithm [11] in our network. Finally, we conclude in Section 5.

2 A Reconfigurable Optical Interconnection Network

We introduce an abstract model for a complete interconnection network [14] using free-space reconfigurable optical interconnects and discuss its characteristics.

Definition. A *reconfigurable optical network*, *RON* (*k*, *N*), consists of *N* computing nodes with their own local memory. A node is capable of connecting directly to any other node. A node can establish *k* simultaneous connections. These connections are established dynamically by reconfiguring the optical interconnect. The links remain established until they are explicitly destroyed.

Messages are sent using *circuit-switching*. Each node has the ability to simultaneously send and receive *k* messages on its *k* links (the *k-port* model), or exactly one message on one of its links (the *single-port* model). A simplified block diagram of the network is shown in Figure 1.

Each node has a fixed number of tunable transmitters and a large number of fixed receivers. We assume an unbounded number of available wavelengths for the system. However, in case of limited number of available wavelengths, one can utilize spread-spectrum techniques where each transmitter sends its information changing the wavelength in a pseudorandom fashion. The receiver can reconstruct the transmitted message if it is aware of the pseudo-random code used for encoding. One can also devise collective algorithms where collision cannot happen at a destination in the network.

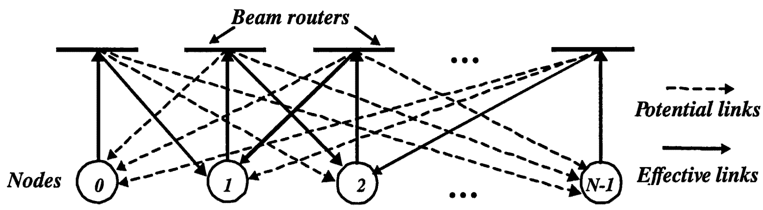

Fig. 1. *RON* (*k*, *N*), a multicomputer interconnected by a complete free-space optical interconnection network.

Various implementation technologies exist to embody the above abstract model. Such technologies include *vertical-cavity surface-emitting lasers* (VCSELs) for photon generation, *self-electro- optic effect devices* (SEEDs) for modulation, frequency hoping for coding, wavelength tuning for transmitters and receivers, *computer generated holograms* (CGH), and *deformable mirrors* (DM) for switching and optical beam routing. However, we are not interested in the technology itself, and implementation concerns are outside the scope of this paper. Instead, we are particularly interested in the abstract model of this network. We shall assume that one or more of the technologies outlined above will be used to implement the proposed interconnect. Under such an implementation, the various overheads associated with the reconfiguration of the network (such as beam steering, setting up the computer-generated holograms, tuning the transmitters, or sending the frequency code in a frequency hoping implementation etc.) are lumped together as the reconfiguration delay *d*.

3 Communication Modeling

Hockney's model [15] characterizes the time for a point-to-point communication as $T = ts + lm\tau$ where ts is the start-up time equal to the time needed to send a zero byte message, lm is the length of message, and τ is the per unit transmission time. For the *RON* (*k*, *N*), we amend the model by explicitly including the reconfiguration delay *d* that is necessary for a node to configure a link that would connect directly to its target node. The transmission time then becomes $T = d + ts + lm\tau$. We incorporate both ts and $lm\tau$ into a single message delay $tm = ts + lm\tau$. Therefore, a unit length message transmission takes $T = d + tm$. For the remaining of the discussion, and without loss of generality, we shall assume that $tm = 1$ for a message size of fixed length.

Culler and his colleagues have proposed the *LogP* model [16] which uses another terminology for communication modeling. LogP models sequences of point-to-point communications of short messages. *L* is the network hardware latency for one-word message transfer. *O* is the combined overhead in processing the message at the sender (o_s) and receiver (o_r). *P* is the number of processors. The gap, *g*, is the minimum time interval between two consecutive message transmission from a processor. Alexandrov and others have proposed the *LogGP* model [17] which incorporates long messages into the LogP model. The Gap per byte for long messages, *G*, is defined as the time per byte for a long message. Bar-Noy and Kipnis have developed the *postal model* [18], a special case of LogP model, where g is one. However, they don't consider the parameters O and G.

A node in LogP, LogGP, and postal models can send another message immediately g time after the previous message has been sent without waiting for the previous message to be delivered at the destination. These models are more suitable for the current state-of-the-art wormhole-routed networks where messages can be pipelined through the network. However, a node in our communication modeling can send another message only after its previous message has been delivered and its link has been reconfigured (if needed). This is because our modeling is a *telephone-like model* based on the circuit-switching technique which is suitable for reconfigurable optical networks.

The model that we have used is slightly different from the model that is offered in [11, 12, 19]. The difference lies in the fact that in the network, *RON (k, N)*, only the sender is allowed to reconfigure, and hence the delay penalties occur there. The receiver, in contrast to the models in [12, 19], and in [11] is entirely passive.

We use the notations B_m for broadcasting time. We derive time complexities of the broadcasting algorithms in the network, *RON (k, N)*, under the model m, where $m \in \{F1, Fk\}$. *F1* stands for full- duplex, single-port communication. While, *Fk* stands for full-duplex, k-port communication.

4 Broadcasting

In broadcasting, a node, assuming node n_0 without loss of generality, sends its unique message to all other nodes. We shall concentrate in techniques that could effectively hide the reconfiguration delay d in the network. By reconfiguration latency hiding, we mean the process in which while some nodes are in their reconfiguration phase, other nodes are in their message transmission phase.

4.1 k-port

The naive algorithm is to let the broadcasting node n_0 inform k new nodes at a step. Clearly, it takes $(d+1)\lceil (N-1)/k \rceil$ time units. In a better algorithm, $B1_{Fk}$, node n_0 sends the message to k other nodes and these k nodes, upon receiving the message, send it to k other distinct nodes each. Continuing this way, the algorithm will terminate after $(d+1)(\lceil \log_k (N(k-1)+1) \rceil - 1)$ time units.

In a more efficient algorithm, $B2_{Fk}$, node n_0 sends the message to k nodes. At the end of this step, k + 1 nodes possess the message which they now send to k other nodes each. Proceeding this way, the algorithm will terminate after $(d+1)\lceil \log_{k+1} N \rceil$ time units.

The above algorithms, $B1_{Fk}$ and $B2_{Fk}$, are logarithmic in time, but they suffer because of the large reconfiguration delay, d, that each node incurs. We are interested in having an algorithm that will overcome the existence of the large reconfiguration delays by essentially hiding it. The algorithm $B1_{Fk}$ can be improved if the configuration of all the links forming the tree proceed in parallel. Hence, in this new algorithm, $B3_{Fk}$, the broadcasting time is $d + \lceil \log_k (N(k-1)+1) \rceil - 1$.

The algorithm, $B2_{Fk}$, can be improved if the configurations take place concurrent to the message transmissions. We adopt a greedy algorithm, $B4_{Fk}$, where a node reconfigures its links to reach k children which lead to a *pre-configured* tree of an appropriate $O(\log_k N)$ depth. As soon as the broadcasting node has finished sending its message, it reconfigures its links to reach another predefined tree. It is understood that while this node is reconfiguring (this takes d steps time units), nodes that have already been configured and are in possession of the message send it to k neighbors each. This process repeats at each node every time it sends the message. This algorithm is optimal since a node after sending/receiving the message immediately reconfigures to send the message to k new nodes. It is similar to the broadcasting algorithm in [11]. Figure 2 depicts the $B4_{Fk}$ algorithm for a 2-port network with 41 nodes and a reconfiguration delay of 1.

It is clear that either this broadcasting network is a dedicated network, or there exists a global control where nodes understand that a broadcasting is going to take place and hence they reconfigure their links correspondingly. In the latter case, an early reconfiguration delay should be added to the broadcasting time.

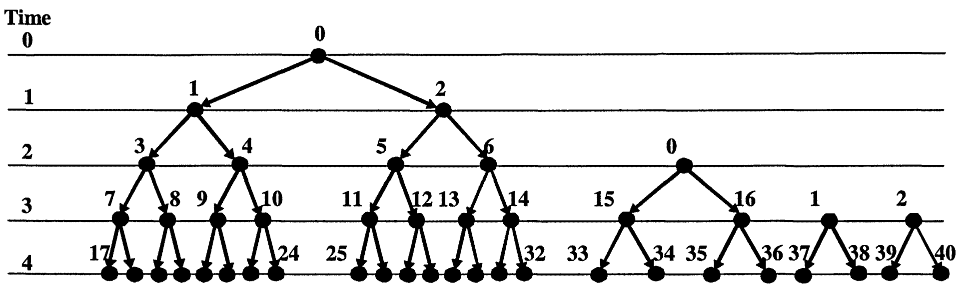

Fig. 2. Latency hiding broadcasting algorithm for RON (k, N), $N =$ 41, $k = 2$, $d = 1$.

4.1.1 *Analysis of the Greedy Algorithm, $B4_{Fk}$*

It is worth mentioning that it can be shown that the total number of nodes, $N(S)$, informed up to step S follows the Eq. (1). It can also be shown that the number of nodes, $r(S)$, that receive the message at each step, S, follows the Eq. (2). These recurrence relations are a kind of generalization of the Fibonacci functions defined in [18], and are similar to the recurrence relations of the broadcasting algorithm in [11]. These relations and those in [18, 11] cannot be solved for a general d. They should be computed step by step or be given in a table in order to find the termination time of the algorithms. However, as will be shown in the following, our analysis of the broadcasting algorithm includes a closed formulation that yields the termination time.

$$N(S) = \begin{cases} 1 & \text{for} \quad S = 0 \\ kN(S-1)+1 & \text{for} \quad S \le d+1 \\ kN(S-1)+N(S-d-1) & \text{for} \quad S > d+1 \end{cases} \tag{1}$$

$$r(S) = \begin{cases} kN(S-1) & \text{for} \quad S \le d+1 \\ kN(S-1)+N(S-d-1) & \text{for} \quad S > d+1 \end{cases} \tag{2}$$

We shall approach the analysis constructively, that is, we shall find the number of nodes that will be informed as time progresses, and we shall stop when all nodes N have been informed. Denote by S the termination time (in units of t_m). Then starting from an arbitrary node n_0, the nodes that will be informed and assuming no reconfiguration, belong to a k-ary tree rooted at node n_0 and of depth S. There are $N_1 = (k^{S+1} - 1)/(k-1)$ nodes in this tree, and we shall reference them as belonging to the first generation. Each of the nodes in this tree, once it has broadcast the message to its own children, will reconfigure and will become the root of a new tree over which a new wave of broadcasting will commence and proceed concurrently with the broadcasting in the first generation tree. This can only happen if $S \geq d+2$ ensuring that the first node to be reconfigured (node n_0) will have enough time to reconfigure and broadcast to its k children.

We shall refer to the nodes belonging to the trees rooted at nodes which were included in the first generation tree and reconfigured, as the second generation nodes. Thus, node n_0 can send its message again at time $d + 1$. By sending this message, n_0 actually embeds a new k-ary tree at depth $d + 1$. The next k nodes at depth 1 of the first generation of trees embed k different k-ary trees at depth $d + 2$. Using this concept, the k^{S-d-2} nodes at depth $S - d - 2$ of the first generation embed the last k^{S-d-2} different trees at depth $S - 1$ in the second generation. Figure 3 depicts the embedding of the first two generations of the nodes.

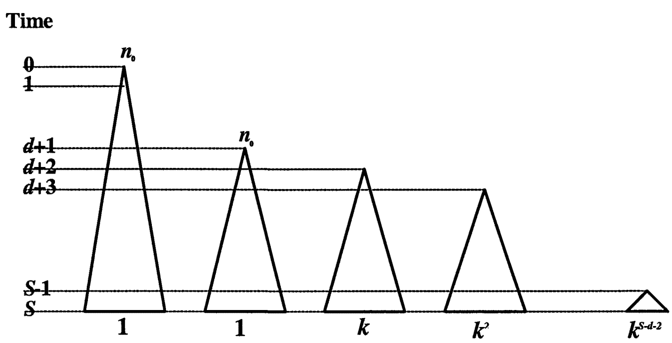

Fig. 3. First and second generation trees. The numbers underneath each tree denote the number of trees having the same height. These trees are rooted at nodes that were at the same level in the first generation tree.

Denote by N_2 the total number of new nodes in the second generation, and by M_i the total number of new nodes in the trees of the second generation rooted at depth i. Therefore,

$$M_{d+1} = k + k^2 + \ldots + k^{S-(d+1)} = \frac{k^{S-d} - k}{k-1}, M_{d+2} = k(k + k^2 + \ldots + k^{S-(d+2)}) = \frac{k^{S-d} - k^2}{k-1},$$

and $M_{d+3} = k^2(k + k^2 + \ldots + k^{S-(d+3)}) = \dfrac{k^{S-d} - k^3}{k-1}$. This continues until depth $S - 1$

where $M_{S-1} = k^{S-d-2}(k) = \frac{k^{S-d} - k^{S-d-1}}{k-1}$. Therefore, the total number of new nodes in the second generation, N_2, will be $N_2 = \sum_{i=d+1}^{S-1} M_i = \sum_{j=1}^{S-d-1} \frac{k^{S-d} - k^j}{k-1}$.

Proceeding the same way as computing N_2, the total number of the new nodes in the third generation N_3, the fourth generation, N_4, and the fifth generation, N_5, will be $N_3 = \sum_{j=1}^{S-2d-2} j(\frac{k^{S-2d-1} - k^j}{k-1})$, $N_4 = \sum_{j=1}^{S-3d-3} \frac{j(j+1)}{2!}(\frac{k^{S-3d-2} - k^j}{k-1})$, and

$N_5 = \sum_{j=1}^{S-4d-4} \frac{j(j+1)(j+2)}{2!} \frac{k^{S-4d-3} - k^j}{k-1}$, respectively.

Lemma 1. The number of new nodes in generation $i + 1$, i • 1 can be found as:

$$N_{i+1} = \sum_{j=1}^{S-i(d+1)} \binom{j+i-2}{i-1} \frac{k^{S-i(d+1)+1} - k^j}{k-1} \tag{3}$$

Proof. We give a combinatorial argument for its validity. Assume a tree belonging to generation i -1 and rooted at depth (i -1)(d + 1). This tree will produce a number of trees belonging to generation i and rooted at depth $i(d + 1)$. The term $(k^{S-i(d+1)+1} - k^j)/(k-1)$ represents the number of new nodes in the first tree of generation i rooted at depth $i(d + 1)$. Subsequent trees in this generation, have a decreasing (by one) number of levels, but since they were produced by nodes that are at lower levels in the parent generation, their numbers grow with the power of k. Therefore, the number of nodes within all the trees at each level, remains the same and equal to $(k^{S-i(d+1)+1} - k^j)/(k-1)$. We have just accounted for the number of trees produced by a single tree in a parent generation. There are though more than one trees of identical depth in the parent generation, and the multiplicative term $\binom{j+i-2}{i-1}$ accounts for this number based on the pascal's triangles [20]. •

Thus, the total number of nodes, $N(S)$, is equal to $N(S) = N_1 + N_2 + ... + N_{\lceil \frac{S}{d+1} \rceil -1}$, or

$$N(S) = \frac{k^{S+1} - 1}{k-1} + \sum_{i=1}^{\lceil \frac{S}{d+1} \rceil -1} \sum_{j=1}^{S-i(d+1)} \binom{j+i-2}{i-1} \frac{k^{S-i(d+1)+1} - k^j}{k-1} \tag{4}$$

Note that Eq. (4) is a closed formula and easier to compute (less computation and memory requirements) than the Eq. (1), and Eq. (2). Table 1 provides a comparison of some numerical examples for the broadcasting time under different broadcasting algorithms, $B1_{Fk}$, $B2_{Fk}$, $B3_{Fk}$ and $B4_{Fk}$, for a particular number of nodes, N, reconfiguration delay, d, and port modeling, k. It is clear that the latency hiding algorithm, $B4_{Fk}$, performs better than the other algorithms.

Table 1. Broadcasting time, $k = 2$

N	d	$B1_{Fk}$	$B2_{Fk}$	$B3_{Fk}$	$B4_{Fk}$
1393	1	20	14	11	8
2703	3	44	32	14	10
2145	5	66	42	16	10
8193	10	143	99	23	12

4.2 Single-port

In this case, a node can only use one of its links. Therefore, instead of k-ary trees, linear arrays are embedded. Hence, using the same concept as in the k-port modeling, the total number of nodes for generations one, two, three, and four are:

$N_1 = S+1$, $N_2 = \sum_{j=1}^{S-d-1} S-d-j$, $N_3 = \sum_{j=1}^{S-2(d+1)} j(S-2d-1-j)$, and $N_4 = \sum_{j=1}^{S-3(d+1)} \frac{j(j+1)}{2!}(S-3d-2-j)$, respectively. If we continue in a similar manner, then the total number of nodes in all generations, $N(S)$, would be:

$$N(S) = S+1+\sum_{i=1}^{\left\lceil \frac{S}{d+1} \right\rceil -1} \sum_{j=1}^{S-i(d+1)} \binom{j+i-2}{i-1}(S-i(d+1)+1-j) \qquad (5)$$

Table 2 provides a comparison of some numerical examples for the broadcasting time of the latency hiding algorithm, B_{F1}, and of the *spanning binomial algorithm* [21], $(d+1)\lceil \log_2 N \rceil$ for a particuar number of nodes, N, and reconfiguration delay, d. It is evident that the algorithm, B_{F1}, performs better than the spanning binomial algorithm.

Table 2. Broadcasting time, $d = 3$

N	$(d+1)\lceil \log_2 N \rceil$	B_{F1}
69	28	12
1252	44	21
8657	56	27
82629	68	34

4.3 Grouping schema

The total number of nodes, $N(S)$, informed up to step S is given as Eq. (1). Meanwhile, the number of nodes, $r(S)$, that receive the message at each step S is defined as Eq. (2). The nodes are divided into two groups. The group that has already received the message and the one that has not. The nodes that know the message at any given step can be grouped into those nodes that have already received the message and those that receive at this time step. The nodes that receive at each step, is proportional (k times) to the number of nodes that have received the message at the last step and those that have sent the message $d+1$ steps ago.

The same grouping schema as in [11] can be used to find the set of nodes that transmit the message at step S, $T(S)$, and the set of the nodes that receive the message at step S, $R(S)$. These two sets can be found by Eq. (6).

$$\begin{cases} T(0) = 0 \\ R(S) = \{N(S-1)+1, \ldots, N(S)\} \\ T(S) = T(S-d-1) \cup R(S-1) \end{cases} \tag{6}$$

5 Conclusion

Free-space optical interconnects provide attractive alternative ways to achieve high speed communications in a multicomputer system. In this paper, we introduced a reconfigurable complete free-space optical interconnection network, *RON (k, N)*. In order to benefit from such interconnects effectively, reconfiguration delay should be hidden.

We presented and analyzed a broadcasting algorithm [11] that could effectively hide the reconfiguration delay d in the network, *RON (k, N)*. In this algorithm, the reconfiguration phase of the nodes is overlapped with the message transmission phase of the other nodes. Our new analysis of the broadcasting algorithm includes closed formulations that yield the termination time.

It is interesting to devise an optimal algorithm for multi-broadcasting such that messages are pipelined in the embedded trees using the latency hiding broadcasting algorithms ($B4_{Fk}$, or B_{FI}). Moreover, although very challenging, efficient algorithms for multicasting, scattering, all-to-all broadcasting, and total exchange should be devised such that they could use latency hiding technique to hide the reconfiguration delay in the network.

References

[1] K. Hwang and Z. Xu, *Scalable Parallel Computing: Parallelism, Scalability, Programmability* (McGraw-Hill, 1998).

[2] J. Duato, S. Yalamanchili and L. Ni, *Interconnection Networks: An Engineering Approach*, (IEEE Computer Society Press, 1997).

[3] R. A. Nordin, A. F. Levi, R. N. Nottenburg, J. O'Gorman, T. Tanbun-Ek, and R. A. Logan, A System Perspective on Digital Interconnection Technology, *IEEE J. Lightwave Technology*, **10** (1992) 801-827.

[4] F. E. Kiamilev, Performance Comparison between Optoelectronic and VLSI Multistage

Interconnection Networks, *IEEE J. Lightwave Technology*, **9** (1991) 1674-1692.

[5] K. Li, Y. Pan and S. Q. Zheng, eds., *Parallel Computing Using Optical Interconnections*, (Kluwer Academic Publishers, 1998).

[6] P. Berthome and A. Ferreira, eds., *Optical Interconnections and Parallel Processing: Trends at the Interface*, (Kluwer Academic Publishers, 1998).

[7] V. Kumar, A. Grama, A. Gupta and G. Karypis, *Introduction to Parallel Computing: Design and Analysis of Algorithms*, (The Benjamin/Cummings Publishing Company, Inc., 1994).

[8] *Message Passing Interface Forum: MPI: A Message-Passing Interface Standard.* Version 1.1.

[9] P. K. McKinley and D. F. Robinson, Collective Communication in Wormhole-Routed Massively Parallel Computers, *IEEE Computer*, December 1995, 39-50.

[10] P. Fraigniaud and E. Lazard, Methods and Problems of Communication in Usual Networks, *Discrete Applied Mathematics*, **53** (1994) 79-133.

[11] P. Berthome and A. Ferreira, Communication Issues in Parallel Systems with Optical Interconnections, *Int. J. Foundations of Computer Science,* **8(2)** (1997) 143-162.

[12] P. Berthome and A. Ferreira, On Broadcasting Schemes in Restricted Optical Passive Star Systems, Interconnection Networks and Mapping and Scheduling Parallel Computations, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, **21** (1995) 19-29.

[13] G. Gravenstreter and R. G. Melhem, Realizing Common Communication Patterns in Partitioned Optical Passive Stars (POPS) Networks, *IEEE Trans. Comput.*, **47(9)** (1998) 998-1013.

[14] A. Afsahi and N. J. Dimopoulos, Collective Communications on a Reconfigurable Optical Interconnect, in *Proc. OPODIS'97, Int. Conf. on Principles of Distributed Systems*, December 1997, 167-181.

[15] R. W. Hockney, The Communication Challenge for MPP: Intel Paragon and Meiko CS-2, *Parallel Computing*, **20(3)** (1994) 389-398.

[16] D. E. Culler, R. M. Karp, D. A. Patterson, A. Sahay, K. E. Schauser, E. Santos, R. Subramonian and T. von Eicken, LogP: Towards a Realistic Model of Parallel Computation, in *Proc. 4th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 1993.

[17] A. Alexandrov, M. Ionescu, K. E. Schauser and C. Scheiman, LogGP: Incorporating Long Messages Into the LogP Model - One Step Closer Towards a Realistic Model for Parallel Computation, in *Proc.* 7th *Annual ACM Symposium on Parallel Algorithms and Architecture*, July 1995.

[18] A. Bar-Noy and S. Kipnis, Designing Broadcasting Algorithms in the Postal Model for Message-Passing Systems, in *Proc. 4th Annual ACM Symposium on Parallel Algorithms and Architectures*, June 1992, 11-22.

[19] F. Desprez, A. Ferriera and B. Tourancheau, Efficient Communication Operations on Passive Optical Star Networks, in *Proc. 1st Int. Conf. on Massively Parallel Processing using Optical Interconnections*, 1994, 52-58.

[20] P. J. Cameron, *Combinatorics: Topics, Techniques, Algorithms*, (Cambridge University Press, 1994).

[21] H. Sullivan and T. R. Bashkow, A Large Scale, Homogeneous, Fully Distributed Parallel Machine, in *Proc. 4th Annual Symposium on Computer Architecture*, March 1977, 105-124.