High Performance in Tree-based Parallel Architectures

Fabio Ancona, Stefano Rovetta and Rodolfo Zunino
D.I.B.E. - University of Genoa - Via all'Opera Pia 11a - 16145 Genova - Italy

Abstract

The integrated schema called "complex node" couples twin processors, which share a dual-port memory supporting a bidirectional, high-speed communication link. The paper considers the contribution of this device to tree-structured processor hierarchies. As shown by theoretical analysis, increased complex node performance may ensure optimality under reasonable conditions in terms of connectivity and memory speed, which can be easily attained by commercial equipment. This endows the overall research with additional value from a technical point of view. The integrated approach has been implemented by using transputers for experimental development; related advantages in a specific application (implementation of associative models) are also highlighted.

1. Introduction

State of the art studies on tree topologies deal with structural properties of parallel architectures handling high-dimensional data [1-4], fault-tolerant properties in binary trees [5] and with the implementation of large binary trees in VLSI [6]. This class of topologies is usefully adopted in many applications, including, for example, FFTs computation [7], and the implementation of associative [8] and neural models [9-11] for high-dimensional data processing. The model showed in this paper aims to stress a system's performance by focusing on connectivity in massive-communication situations where an effective control of information flow is crucial.

A methodology is here presented to enhance the performance of tree-based parallel architectures. To this end, a novel integrated device, the *complex node*, is illustrated which couples two twin processors (featured by communication links) through a shared dual-port memory, thus obtaining double connectivity and a double computational power. This shared memory, with the support of software primitives, can be considered as an additional high-speed virtual connection. Complex node throughput is increased by an efficient strategy for optimization of device resources. An impact analysis of this integrated component in hierarchical architectures is presented, proving the effective gain in overall system performance. An important result obtained by this research is that the complex node achieves optimal performance by using low connectivity processors. This result is also interesting from a technical perspective, as commercially available processors can easily meet optimality conditions.

Hardware implementation of the proposed method involved transputer-based architectures [12], which were selected for their satisfactory connectivity. In addition, transputers offer an effective and relatively inexpensive support to further development. The experimental validation of the presented research shows that the actual complex device implementation meets the basic timing conditions for ensuring optimality. The paper also discusses the advantages gained from including the higher-connectivity structure in a tree hierarchy for supporting associative models. An analytical expression of ultimate system efficiency was derived in a previous research [8].

2. Structural Analysis

2.1. Basic Complex node architecture

The overall approach models a general family of processors for a general class of problems and is not restricted to any specific hardware machinery. The model of the node is intended to maintain the highest level of generality without practical reference to the supporting circuitry.

In the complex node (Fig.1), two processors share a dual-port memory, which they can access independently. The processor model assumes, without loss of generality, I/O bidirectional links, supporting a processor connectivity greater than or equal to 2. Both processors in the complex node share a common address bus and a data bus to access

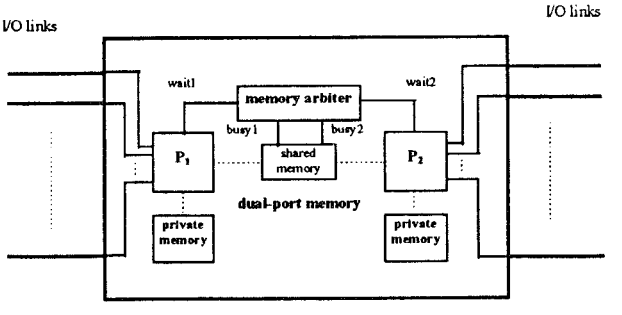


Fig. 1 - General architectural schema of the complex node

1089-6503/97 $10.00 © 1997 IEEE

their private memory or the shared memory. Memory conflicts occur when both processors access the same memory location and one of them executes a write cycle. A simple memory controller solves these conflicts through memory access coordination and tracking of processor requests.

The crucial idea underlying the processor coupling is to develop a device overcoming the limited connectivity provided by one elementary component. This enhancement may have a relevant impact on the effectiveness of a parallel system, since it virtually doubles the connectivity of a basic element, while maintaining overall compatibility. For example, a direct advantage resulting from this improvement is in the possibility of building higher-dimensional hypercubes [13] otherwise prevented by processor structural limitations. The shared memory operates as an additional "virtual link". Owing to its bidirectional nature, the higher internal bus access speed is used to accelerate the information transfer among node constituents.

2.2. General model for structural analysis

Hierarchical computing architectures typically support problems in which computational cost plays a key role. The crucial issue in this class of architectures is effective information flow control, especially when communication flows uniformly in one direction (either top-down or bottom-up). In other words, efficiency critically depends on how fast each node spreads data from the upper to the lower levels; increasing "diffusion" performance ultimately decreases communication overheads.

Therefore, the theoretical model analyzes the complex node when operating as a "diffusing" unit for top-down data flow. Thus the study focuses on the role of the increased connectivity in tree topologies.

Structural analysis follows a comparative approach and concentrates on two basic schemata that use elementary processors and have different internal structures. The first schema considered for comparison includes only one processor and is taken as the simplest reference for the information-diffusing task. The second schema is more similar to the complex-node model; it is compared with a dual structure connecting two elementary units by a standard link (and not by a shared-memory bank).

The total branching factor varies in the different schemata. The theoretical analysis defines a homogeneous parameter to attain significant comparisons, namely, effective communication speed. This quantity is measured in bits/sec and models the structure effectiveness as a diffusing unit. In this way measurements are independent of the different total connectivity of each configuration. Without loss of generality, data flow assumed to be uniformly distributed within the network: the data received by each processor are proportional to the number of "downward" output links. Figure 2 illustrates the different configurations examined, where P denotes the data packet dispatched to each processor at the lower level, and c indicates elementary connectivity. Two different cases are considered for a more general approach to the analysis:

A) total time includes only the time required to transfer data to the lower nodes; this implies that data are already present in the node when downward transmission starts; otherwise:

B) total time is the sum of data receiving and data diffusion time.

The analysis compares complex-node performance with the timing of each reference schema; data-transmission speed (bits per second) is worked out for each configuration. Finally, a *gain* function (the ratio of the evaluated speed values) quantifies the relative improvement of the complex node. All these quantities are expressed as functions of connectivity, c, of elementary processor, and, for the complex node, as functions of link speed-up, S.

A final remark on the specific communication process model: communications proceed logically through links as parallel processes; actually, processors often handle communications sequentially owing to the impossibility to perform parallel memory accesses. The model for communication timing adopted accounts for this limitation; yet, the analysis applies also to processors supporting truly parallel communications: both the complex node and elementary-processor schemata will benefit from this feature to the same extent. The following notations will be adopted:

- P = data packet size (no. of bits) dispatched to each node at the lower level;
- P_{TOT} = total amount of data (bits) to be transmitted;
- n_{OUT} = number of "downward" output links from a node (branching factor);
- τ_p = time for transferring a packet through a physical link;
- S = speed-up provided by the shared-memory link as compared with the physical link:

$$S = \frac{Speed_{virtual-link}}{Speed_{physical-link}}$$

- T_{IN} = total time to receive the data to be transmitted;
- T_{OUT} = total time to transmit data;
- b_x = overall data-diffusion speed, [bit/sec], for each

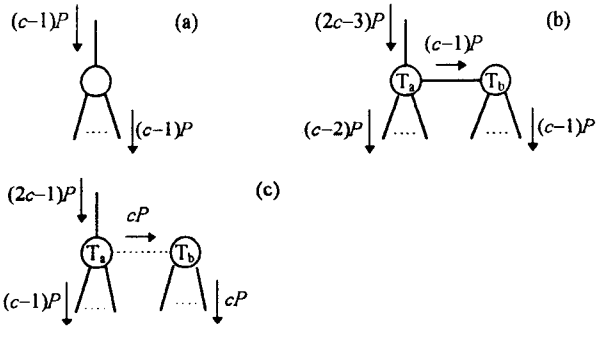


Fig. 2 - Structures used for comparisons.
a) Single Processor (S); b) Two processors coupled by a physical link (D) ; c) Complex node (N)

configuration in Fig. 2 { a, b, c }: x = { S, D, N }, respectively.

2.3. Case A: data-receiving time not included

In this case, the total diffusion process involves only the time to transfer data to the lower nodes. For each diffusor, the amount of data to be transmitted, P_{TOT} , is equal to the packet size multilpied by the number of output links from the diffusor node; P_{TOT} *to* T_{OUT} ratio gives the data-diffusion speed, $b_x^{(A)}, x \in \{S, D, N\}$.

Single-processor schema (Fig. 2a): in this case, both the total amount of handled data and related diffusion time are proportional to the basic elementary processor connectivity. In this case, data diffusion speed coincides with the speed of a physical link.

$$P_{TOT} = n_{OUT} \cdot P = (c-1) \cdot P$$
$$T_{OUT} = (c-1) \cdot \tau_p$$
$$b_S^{(A)} = \frac{P_{TOT}}{T_{OUT}} = \frac{(c-1) \cdot P}{(c-1) \cdot \tau_p} = \frac{P}{\tau_p} \qquad (1)$$

Pair of processors coupled through a physical link (Fig. 2b): This structure increases connectivity and is similar to a complex node because it has the same computational power. The structure is not symmetrical, as processor T_a uses c–2 links for data diffusion, whereas T_b uses c-1 links. Processors can send data to the lower layer simultaneously; a specific transmission schedule can partially overcome the link-based communication drawback:

- T_a transmits c–1 packets to T_b through the physical link; the cost of this phase is proportional to c-1;
- the two processors perform (simultaneous) downward data transmissions; the cost of this phase is ruled by transmission timing for T_b (which handles the largest data packet), and is proportional to c-1.

As a result, the quantities concerned in the comparison can be expressed as:

$$P_{TOT} = n_{OUT} \cdot P = (2c-3) \cdot P$$
$$T_{OUT} = (c-1) \cdot \tau_p + (c-1) \cdot \tau_p$$

Down Transm. from Tb

Transm. from Ta to Tb

$$b_D^{(A)} = \frac{P_{TOT}}{T_{OUT}} = \frac{2c-3}{2c-2} \cdot \frac{P}{\tau_p} \qquad (2)$$

Complex node (Fig. 2c): the internal complex node structure enhances two basic features of a system performance. First, the shared-memory interconnection increases message-passing speed between the two node processors; second, the additional virtual link increases the number of connections available for tree connectivity (i.e., the number of output links increases up to 2c-1). The intra-node communication schedule follows the same strategy previously adopted for case of figure 2b:

$$P_{TOT} = n_{OUT} \cdot P = (2c-1) \cdot P$$
$$T_{OUT} = c \cdot \frac{1}{S} \cdot \tau_P + c \cdot \tau_P$$

Down Transm. from Tb

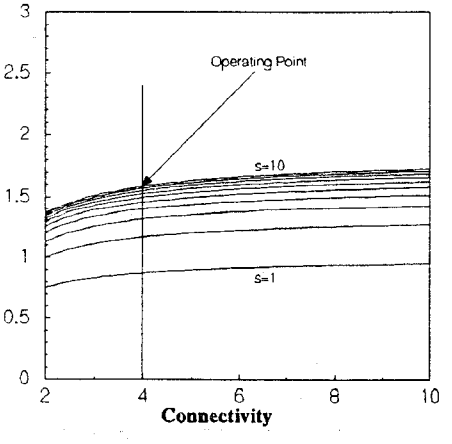
Transm. from Ta to Tb

$$b_N^{(A)} = \frac{P_{TOT}}{T_{OUT}} = \frac{2c-1}{c \cdot (1 + \frac{1}{S})} \cdot \frac{P}{\tau_P} \qquad (3)$$

2.4. Case B: analysis including data-receiving time

This case takes into account also the time required by the diffusing unit to receive data. This analysis is motivated by the fact that the various schemata have different total connectivities, hence the different amounts of handled data might affect the overall throughput. The analysis is very similar to the one presented in case A).

Single-processor schema (Fig. 2a): Total time for data handling is twice as much as the analogous configuration in case A), half for input and half for output, which ultimately



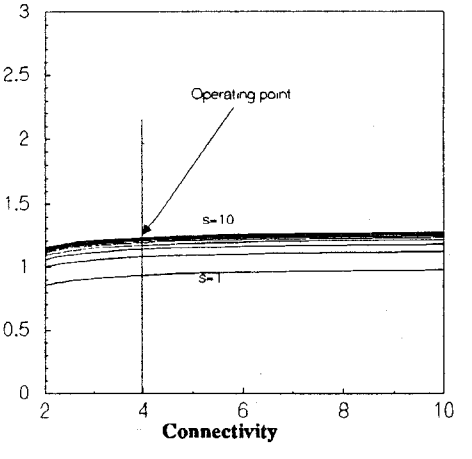


Fig.3 - Gain curves computed with respect to a single-processor diffusor. The arrows indicate the operating point of a real technical implementation (Case A) Curves not including T_{IN} in computation, Case B) Curves including T_{IN} in computation)

halves diffusion speed, due to sequential communications.

$$T_{TOT} = T_{IN} + T_{OUT} = 2 \cdot (c-1) \cdot \tau_P$$
$$b_S^{(B)} = \frac{P_{TOT}}{T_{TOT}} = \frac{c-1}{2 \cdot (c-1)} \cdot \frac{P}{\tau_P} = \frac{1}{2} \cdot \frac{P}{\tau_P} \quad (4)$$

Pair of processors coupled through a physical link (Fig. 2b): in this situation, additional data receipt time is proportional to the branching factor (i.e., 2*c*-3).

$$P_{TOT} = n_{OUT} \cdot P = (2c-3) \cdot P$$
$$T_{TOT} = T_{IN} + T_{OUT} = (2c-3) \cdot \tau_P + (2c-2) \cdot \tau_P = (4c-5) \cdot \tau_P$$
$$b_D^{(B)} = \frac{P_{TOT}}{T_{TOT}} = \frac{2c-3}{4c-5} \cdot \frac{P}{\tau_P} \quad (5)$$

Complex node (Fig. 2c): in this case, too, one of the two processors accounts for additional data receipt cost for receiving data:

$$P_{TOT} = n_{OUT} \cdot P = (2c-1) \cdot P$$
$$T_{TOT} = T_{IN} + T_{OUT} = (2c-1) \cdot \tau_P + c \cdot (1 + \frac{1}{S}) \cdot \tau_P = [(3 + \frac{1}{S}) \cdot c - 1] \cdot \tau_P$$
$$b_N^{(B)} = \frac{P_{TOT}}{T_{TOT}} = \frac{2c-1}{c \cdot (3 + \frac{1}{S}) - 1} \cdot \frac{P}{\tau_P} \quad (6)$$

2.5 - Gain analysis

When comparing the complex node with a single processor, the analytical expression for relative performance improvement in this case is obtained by combining (1) and (3) for case A), and (4) and (6) for case B):

$$(A)\ \Delta_{S,N}^{(A)} = \frac{b_N^{(A)}}{b_S^{(A)}} = \frac{2c-1}{c \cdot (1 + \frac{1}{S})}$$
$$(B)\ \Delta_{S,N}^{(B)} = \frac{b_N^{(B)}}{b_S^{(B)}} = 2 \cdot \frac{2c-1}{c \cdot (3 + \frac{1}{S}) - 1} \quad (7)$$

Likewise, the gain of the complex node over the pair of coupled processors is derived by dividing (2) by (3) for case A), and (5) by (6) for case B).

$$(A)\ \Delta_{D,N}^{(A)} = \frac{b_N^{(A)}}{b_D^{(A)}} = \frac{4c^2 - 6c + 2}{2 \cdot (1 + \frac{1}{S}) \cdot c^2 - 3 \cdot (1 + \frac{1}{S}) \cdot c}$$
$$(B)\ \Delta_{D,N}^{(B)} = \frac{b_N^{(B)}}{b_D^{(B)}} = \frac{8c^2 - 14c + 5}{2 \cdot (3 + \frac{1}{S}) \cdot c^2 - (11 + \frac{3}{S}) \cdot c + 3} \quad (8)$$

Gain is an important comparative measure since it expresses the relative advantage of the complex node as a function of basic architectural parameters, i.e. processor connectivity and the communication speed-up provided by the memory virtual link.

All gain expressions are greater than unity, thus indicating that the complex node is better, provided that $S > 1$. This trivial result shows that a better complex node performance also results from the structural feature of setting memory-based communication; such improvement increases with faster memories.

The asymptotic behaviour of these curves when processor connectivity increases could be also be investigated. This leads to the evaluation of the (optimal) connectivity yielding the best performance, and might ultimately drive processor-design strategies. Asymptotic performances for the functions derived from (7)-(8) are given by the following expressions:

$$(A)\ \lim_{c \to \infty} \Delta_{S,N}^{(A)} = \frac{2}{1 + \frac{1}{S}} > 1, \quad \forall S > 1 \quad (9)$$
$$(A)\ \lim_{c \to \infty} \Delta_{D,N}^{(A)} = \frac{2}{1 + \frac{1}{S}} > 1, \quad \forall S > 1 \quad (10)$$
$$(B)\ \lim_{c \to \infty} \Delta_{S,N}^{(B)} = \frac{4}{3 + \frac{1}{S}} > 1, \quad \forall S > 1 \quad (11)$$
$$(B)\ \lim_{c \to \infty} \Delta_{D,N}^{(B)} = \frac{4}{3 + \frac{1}{S}} > 1, \quad \forall S > 1 \quad (12)$$

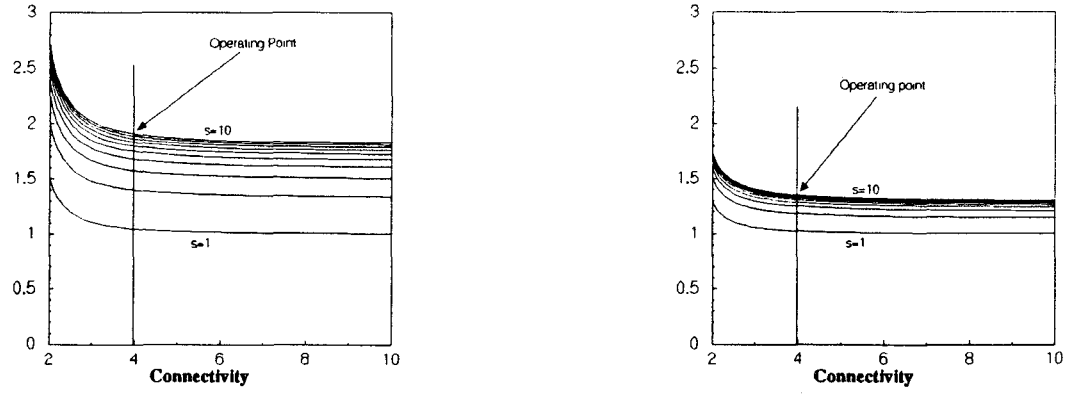


Fig.4 - Gain curves computed with respect to a diffusor composed of two processors coupled hrough a physical link. (A) Curves not including T_{IN} in computation, B) Curves including T_{IN} in computation)

Interestingly, expressions (9-10) and (11-12) show that the limits for cases A and B are equal. This proves that when connectivity increases the configuration with a pair of coupled processors assimilates to the single-processor schema. Another important characteristic of functions (7)-(8) is that each gain improvement depends on the speed-up, S, in a nonlinear fashion. With regard to speed-up, any $S >> 1$ value, in particular $S \geq 10$, makes it possible to attain an asymptotic performance for *any* connectivity. As shown below, this condition on S can be easily fulfilled by commercial equipment. This has important technical consequences on final overall method effectiveness.

The graphs in Fig.3 give the gain curves (7) as compared with the single-processor schema. They also show the operating point achieved by a current technical implementation with $c = 4$. This demonstrates that even inexpensive systems may place the complex-node operating point almost on the asymptote, thus approximating optimality.

The comparison with the two-processor schema appears much more significant, as it considers structures with the same computational power. Obviously, the relative improvement increases when virtual link speed-up increases. However, the graphs in Fig.4 plotting eqns. (8) confirm again that an S=10 value may give a satisfactory asymptotical approximation. The gain is greater than unity also at low connectivities, thus proving that the complex node advantage is a consequence of its own structure and is unrelated to specific technological features.

More importantly, in the graphs, the gain is a *decreasing* function, hence optimal performances are attained at lower connectivities. This seems to be an important result, as it demonstrates that connectivities of commercially available devices support optimal complex nodes (e.g., $c = 4$ for transputers). In other words, with regard to processor hierarchies, redesigning a new processor with higher connectivity is not only far more expensive but even less efficient than coupling two existing ones into a complex structure, as described in this paper.

3. Experimental Results

The general model implementation of the complex node on existing machinery is based on the positive structural analysis results described in the previous Section. This implementation is also feasible for the availability of several processor families with features ensuring the best complex node performance.

In principle, any device with connectivity supporting an MIMD schema is a possible "target" component for making a complex node. Some DSP devices meet these conditions (e.g., Texas Instruments TMS320C40 [14], or Inmos IMSA110 [15]). For the research presented in this paper, transputers [16] have been chosen as the basic constituting elements of the complex architecture. Although other computing platforms can provide superior performance in terms of computational power, the selected devices offer a suitable development environment at affordable costs; in addition, their connectivity (c=4) also meet the optimality requirement previously discussed.

3.1. Hardware implementation

The transputer-based implementation of the complex node [12] includes two T800 processors coupled by means of a dual-port memory; the whole unit has been developed on a standard TRAM PC-bus hosted board to attain maximum flexibility and compatibility with standard transputer-based enviroments.

Each node transputer uses the same address bus to access both its private as well as shared memory; memory selection is triggered by address bus MSB. The private memory section includes 1MB for each processor. A bank of four CY7C132 chips (storing 2K bytes each) supports the dual-port memory link; the memory device embeds an arbiter to solve conflicts, which occur whenever one processor tries a memory access, while the other one is executing a write cycle at the same memory location.

The on-chip arbitration logic cannot keep track of the sequence processor requests, hence an additional, external circuitry rules memory control. Circuitry includes an EPROM storing a finite-state machine; the embedded logic generates appropriate signals for the processor losing a conflict, which is kept in a "wait" state until the other processor releases the contended memory location. The overall conflict-handling algorithm, the associated finite-state machine, and the actual implementation logic are described in detail in [12].

3.2. Improving communication speed-up through shared memory

This section describes an accelerated data-transmission technique that can double intra-node communications, for both mono- and bidirectional data flows. For semplicity and without loss of generality, the analysis addresses the monodirectional case. Applications based on tree architectures mainly involve a synchronous one-way data flow; simultaneous bidirectional communications are mostly infrequent. The acceleration method aims to reduce the waiting time involved in a data exchange between the two processors, and its core involves a straightforward block-switching mechanism:

- the shared memory is split into two equal shares;
- exchanged data are split into equal slices, whose size is equal to half the shared memory;
- the switching mechanism alternates accesses to the two memory halves: when the transmitter writes a data slice into one memory half, the receiver reads another (previously written) slice from the other memory half.

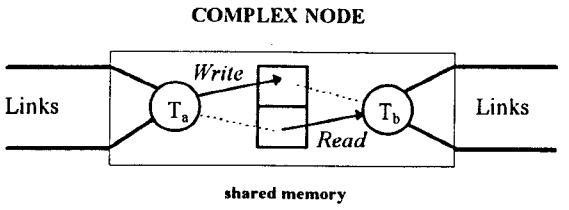


Fig. 5 - The block-switching technique for (one-way) communications inside the complex node.

Figure 5 presents a switch-based communication schematically. Simple derivations show that this technique can virtually double memory-based speed performance. In the following equations, W denotes the number of words in a data packet, M is the size of shared memory, and T_M is the total time for accessing a memory block of size M. Communication time, T_1, through the shared memory *without* block switching is:

$$T_1 = 2 \cdot \frac{W}{M} \cdot T_M \tag{13}$$

where the factor 2 derives from the sequential communications, blocking the transmitter until the reader has released the shared memory bank. Instead, the timing, T_2, *with* block switching is given by:

$$T_2 = \left(\frac{W}{\frac{1}{2} \cdot M} \right) \cdot \left(\frac{1}{2} \cdot T_M \right) + 2 \cdot \frac{M}{2} \cdot T_M = \left(\frac{W}{M} + M \right) \cdot T_M \tag{14}$$

where the first term in the summation represents standard operating timing, whereas the second term includes the two time slots for the initial start-up and the final download. As previously anticipated, expressions (1) and (2) indicate the asymptotic behaviour of the memory speed-up. More precisely, in communication-intensive applications for which one can assume $W >> M$, the above expressions indicate that $T_2 \cong \frac{1}{2} T_1$.

The ratio between the accelerated memory-link speed and transputer physical-link speed gives the speed-up parameter determinining final complex-node performance. As this parameter changes with the amount of exchanged data, measurements covered variable data packets with increasing sizes. In order to maintain unformity with the basic approach followed in the paper, the analysis considers the hardest task of massive data handling, which is most frequent in high-dimentional data processing.

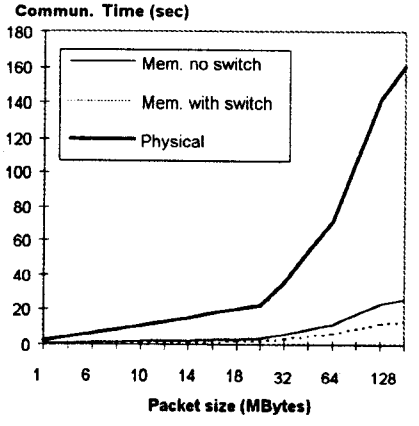


Fig. 6 - Comparative graph of communication timings

Figure 6 gives measured communication timings for the various mentioned configurations (physical link, shared-memory link without and with block switching.). The comparative graph also demonstrates the remarkable performance improvement obtained using the block-switching mechanism.

An interesting feature of experimental data is that the plotted curves exhibit a very similar shape, which indirectly confirms the validity of modelling the shared-memory approach as an additional, high-speed virtual link for intra-processor communication.

3.3. Results on applications involving processor-tree topologies

The experimental testbed selected for verifying the effectiveness of the described structure is an application with massive communication load, involving extensive computations of two-dimensional FFTs of images. The overall testbed is significant from a methodology perspective owing to the high-dimensional nature of the processed signal and the generality of the considered computation.

The application including such an amount of visual signal processing requires the parallel implementation of associative memories for image recognition; the specific framework adopted is the noise-like coding model of associative memory [17]. Massive use of image FFTs derive from the mathematical formalism underlying this model, which involves matrix convolutions and correlations as basic operations [18-19]. Previous research confirmed the relevance and the effectiveness of this memory model for image understanding [20] and integration within stimulus-response devices for real-time systems [21]. This testbed is quite interesting for the current research, since a general methodology has been developed for implementing the memory model on tree-structured parallel architectures. An analytical expression is also available for the overall system efficiency [8]. In this way the contribution of the complex node can be assessed quantitatively and analytically.

The basic approach compares two architectures having the same computational power, the first based on standard transputers and the second including complex nodes. The above-mentioned analysis showed that the efficiency of the tree-structured parallel architecture can be written as

$$\eta = \frac{1}{1 + f(CO)} \tag{15}$$

where f is a monotonically increasing function proportional to the communication cost of the overall system. The term

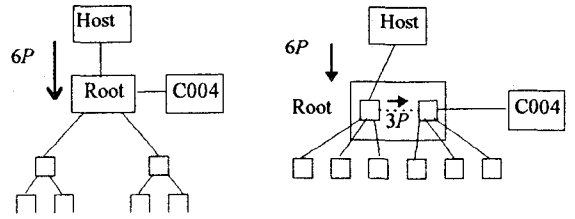


Fig. 7 - Hierachical structures composed of 6 processing units (a) The Root is a transputer, b) The Root is a complex node)

CO accounts for communication overheads and depends only on the *topology* of the considered processor tree. Actually, other terms contribute to determine *f*; such factors, however, can be neglected in the comparative analysis without loss of generality (for instance, they depend on the computational power of involved processors.)

In order to evaluate *CO*, the formalism presented in [8] defines a quantity, denoted *communication-cost factor*, $CF^{(l)}$, associated with the l^{th} level of the hierarchy. This term is defined as:

$$CF^{(l)} = \max_j \{\sum_k D(l,j,k)\} \; ; \quad l=1,\ldots,m \qquad (16)$$

where *m* is the number of levels in the tree, and *D(l, j, k)* is the number of nodes of the subtree spanned from the k^{th} link of the j^{th} processor at level *l*. This formula derives from the total amount of time spent to transmit a data packet through the hierarchy, and takes into account the sequential nature of communications within a transputer. The total communication overhead, *CO*, of the whole architecture is calculated as

$$CO = \sum_{l=0}^{m} CF^{(l)} \qquad (17)$$

The comparative analysis involves only *CO* evaluation for the structures considered, hence the topological

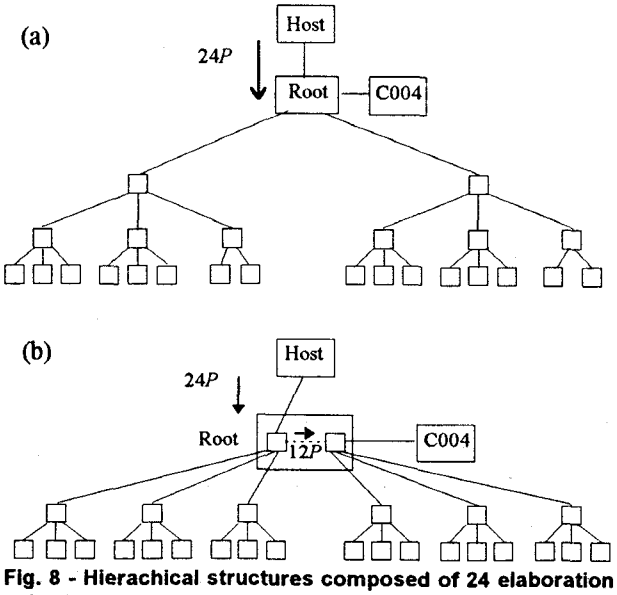


Fig. 8 - Hierachical structures composed of 24 elaboration units (a) The Root is a transputer, b) The Root is a complex node)

improvement obtained through increased connectivity is clearly calculated. All cases consider the most elementary structures, in which the complex node is placed at the root of processor hierarchy. As a matter of fact, theoretical overall architecture analysis indicates that connectivity plays a crucial role when available at the highest level in the hierarchy.

As usual for this family of architectures, the schemata must take into account configuration circuitry; the dedicated board is indicated as C004 and interfaces through a link from the root. Computations always adopt a standard speed-up value of $S = 10$, in compliance with the previously assessed operating point. An important consequence of fast memory speed-up is that, in computation of topological terms (16), all nodes "beyond" the virtual link must be weighted by a factor of $1/S$, as related transmission time is reduced accordingly. In order to be compatibile with the analysis described in [8], and also to isolate connectivity effect on overall performance, the root node is not involved in the actual computations, and operates as a "diffusing" unit as modelled throughout this paper.

The first schema considered implements the simplest possible configuration, namely, processor attached to each link of the complex node. In compliance with the above conventions, the total number of effective processors is six; the corresponding transputer-based architecture with the same computational power requires two levels of processors. Figure 7 presents the related topologies schematically. Communication factors for the presented architectures are determined according to (16) and (17), and are as follows:

Fig. 7a: $D(0, j, k) = 3 \; j=0 \;\; \forall k$; $D(1, j, k) = 1 \; \forall j \; \forall k$; $D(2, j, k) = 0 \; \forall j \; \forall k$;

$CF^{(0)} = 2$;$D(0, j, k) = 6$; $CF^{(1)} = 2 \cdot D(1, j, k) = 2$;

$CF^{(2)} = 0$; $CO_S = 6+2+0 = 8$ (18)

Fig. 7b: $D(0, j, k) = 1 \;\; j=0 \;\; k=1,2,3$; $D(0, j, k) = 1/S$ $j=0$, $k=4,5,6$; $D(1, j, k) = 0 \; \forall j \; \forall k$;

$CF^{(0)} = 3 \cdot 1 + 3 \cdot 1/S = 3.3$; $CF^{(1)} = 0$;

$CO_V = 3.3+0 = 3.3$ (19)

The ratio between these two, quantifies the advantage gained by using the complex node over a standard transputer architecture:

$$\Delta_{eff} = \frac{CO_S}{CO_N} = \frac{8}{3.3} = 2.42 \qquad (20)$$

This analysis demonstrates the remarkable improvement brought about by increased complex node connectivity; the communication-related term in the efficiency formula is reduced by a factor greater than two. It could be argued that this result has been obtained under very simplistic conditions, hence a similar structural analysis will be performed for a more complicated structure with a higher computational power. In particular, the second schema is obtained by adding a full computational level; this generates a processor tree including 24 units. Figure 8 displays the topology for the complex-node architecture and for the equivalent transputer tree.

Fig. 8a: $D(0, j, k) = 12\ \forall j\ \forall k$;
$D(1, j, k) = 4\ \forall j$; $k=1,2$; $D(1, j, k) = 3\ \forall j$; $k=3$;
$D(2, j, k) = 1\ \forall j\ \forall k$; $D(3, j, k) = 0\ \forall j\ \forall k$;
$CF^{(0)} = 2 \cdot D(0, j, k) = 24$;
$CF^{(1)} = 2 \cdot D(1, j, 1) + D(1, j, 3) = 11$;
$CF^{(2)} = 3 \cdot D(2, j, k) = 3$; $CF^{(3)} = 0$;
$CO_S = 24+11+3 = 38$ (21)
Fig. 8b: $D(0, j, 1) = 4$; $j=0$; $k=1,2,3$;
$D(0, j, 1) = 4/S$; $j=0$; $k=4,5,6$;
$D(1, j, k) = 1\ \forall j\ \forall k$; $D(2, j, k) = 0\ \forall j\ \forall k$;
$CF^{(0)} = 3 \cdot 4+3 \cdot 4/S = 13.2$;
$CF^{(1)} = 3 \cdot D(1, j, k) = 3$; $CF^{(2)} = 0$;
$CO_V = 13.2+3+0 = 16.2$ (22)

Thus the differential gain for the architectures shown in Fig. 8 is calculated as follows:

$$\Delta_{eff} = \frac{CO_S}{CO_N} = \frac{38}{16.2} = 2.35 \quad (23)$$

showing that the complex node contribution remains almost unaffected for larger structures including many processors. In other words, the differential gain in efficiency remains greater than two.

This analysis proves the advantages of the complex node in terms of connectivity and throughput; it is worth stressing that the obtained results hold independently of the specific processors involved, and are a consequence of enhanced node structure features; more importantly, the theoretical nature of this analysis aims to evaluate, besides performance gain in terms of communication cost, the generality of the approach proposed. This result is unrelated to the specific application considered, but is valid for any application involving uniform data allocation within a hierarchical processor organization.

IV. Conclusions

A general application implemented on a tree-based structure increases its performance when it includes complex nodes, as they increase throughput performance of the overall system. This novel integrated component combines a pair of processors by means of a dual-port memory operating as an additional, high-speed virtual connection.

This paper first analyses, from a theoretical point of view, the basic idea of coupling processors which can attain higher performance when applied to a general class of systems: hierachical architectures for data processing. This theoretical analysis has demonstrated the novel device's performance using a function depending on two structural parameters, that is, processor connectivity and memory-link speed-up. Two major results have been obtained by this research: the best performance is attained using low-connectivity components, which are commercially available (i.e. transputers) at an affordable price; best results are independent on technological features of the component used. The significant results obtained are likely to drive more efforts to develope new processor families, for which computational power is a much more significant feature than basic connectivity.

Then, the paper examines complex node contribution in the hierarchical architectures quantitatively and analytically. Through theoretical analysis, tree structures based on complex nodes have shown a significant performance gains in terms of communication cost, if compared to the architectures based on standard processors. The generality of the approach proposed has also ensured and is valid for any application involving a uniform data allocation in tree-based parallel architectures.

References

[1] Q. M. Malluhi and M.A. Bayoumi, The hierarchical hypercube: a new interconnection topology for massively parallel systems, *IEEE Trans. on Parallel and Distr. Sys.* 5 (1) (1994) 17-30.

[2] J. F. Jenq and S. Sahni, Image shrinking and expanding on a pyramid, *IEEE Trans. on Parallel and Distr. Sys.* 4 (11) (1993) 1291-1296.

[3] S. L. Scott and G. S. Sohi, The use of feedback in multiprocessors and its application to tree saturation control, *IEEE Trans. on Parallel and Distr. Sys.* 1 (4) (1990) 385-398.

[4] V. Bharadwaj, D. Ghose and V. Mani, Optimal sequencing and arrangement in distributed single-level tree networks with communication delays, *IEEE Trans. on Parallel and Distr. Sys.* 5 (9) (1994) 968-976.

[5] M.Y. Chan and S.J. Lee, Fault-tolerant embedding of complete binary trees in hypercubes, *IEEE Trans. on Parallel and Distr. Sys.*, 4 (3) (1993) 277-288.

[6] H.Y. Youn and A.D. Singh, On implementing large binary tree architectures in VLSI and WSI, *IEEE Trans. on Computers*, 38 (4) (1989) 526-537.

[7] Y. Huang and Y. Parker, A parallel FFT algorithm for transputer networks, *Parallel Computing* 17 (1991) 895-906.

[8] F. Pagano, GC. Parodi and R. Zunino, Parallel implementation of associative memories for image classification, *Parallel Computing* 19 (6) (1993) 667-684.

[9] G.W. Cottrell, P.Munro and D. Zipser, Learning internal representation from gray-scale images: an example of extansional programming, *Proc. 9 Annual Cogn. Sc. Soc. Congress, Wa. 1987.*

[10] D. Anguita, F. Passaggio and R. Zunino, SOM-based interpolation to image compression, *Proc. World Congress on Neural Networks WCNN'95*, July 1995, vol. I, pp. 739-742.

[11] S. Ridella, S. Rovetta and R. Zunino, Generalization-based approach to plastic vector quantization, *Proc. World Congress on Neural Networks WCNN'95*, July 1995, vol. I, pp. 505-508.

[12] F. Ancona, D. Anguita, S. Rovetta and R. Zunino, Shared-memory architecture to implement a high-connectivity processing node, *Electronics Letters* 31 (22) (1995) 1903-1904.

[13] Y. Saad and M. H. Schultz, Topological properties of hypercubes, *IEEE Trans. on Computers* 37 (7) (1988) 867-872.

[14] Texas Instruments, *TMS320C40 User's Guide* (Dallas, 1991).

[15] SGS-Thomson, *The Digital Signal Processing databook* (1989).

[16] INMOS, *The Transputer databook* (1989).

[17] S. Bottini, An algebraic model of an associative noise-like coding memory, *Biol. Cybern.* 36 (1980) 221-228.

[18] D. Gabor, Holographic model of temporal recall, *Nature (London)* 217 (1968) 548.

[19] D. Gabor, Improved holographic model of temporal recall, *Nature (London)* 217 (1968) 1288-1289.

[20] A. Diaspro, GC. Parodi and R. Zunino, A performance analysis of an associative system for image classification, *Pattern Recognition Letters* 14 (1993) 861-868.

[21] D. Anguita, GC. Parodi and R. Zunino, Associative Structures for Vision, *Multidimensional Systems and Signal Processing* 5 (1994) 75-96.