計算機アーキテクチャ　91−13
（１９９１．１１．２２）

MANDALA:超並列計算機向き相互結合網

FLAVELL Andrew　加納 卓也　藤本 茂訓　高橋 義造
徳島大学工学部知能情報工学科

超並列計算機の構成に適した拡張性のある階層構造結合網としてＭＡＮＤＡＬＡを提案する．この結合網は完全結合網を再帰的に結合することによって構成される階層構造結合網である．すなわちC個のプロセッサ要素を完全結合してレベル１のクラスタを構成し，さらにこのクラスタをＣ個完全結合してレベル２のクラスタを構成することを繰り返し，最終的にレベルLのクラスタを構成する．このようにしてノード数$N=C^L$，次数C，直径$D(L)=2^L$-1のレベルLのＭＡＮＤＡＬＡ結合網が構成される．この結合網の平均通信距離はクラスタサイズとレベル数の関数となり，$O(N^{1/log_2C})$となる．また$L-1$レベルの中継ノードのメッセージ密度は$O(N^2)$となる．中継ノードは一様であるので拡張性に優れている．メッセージの中継と放送のためのアルゴリズムは単純になることを示し，また障害時の中継方式についても検討している．

MANDALA: An Interconnection Network for a Massively Parallel Computer

Andrew FLAVELL, Takuya KANOH, Shigenori FUJIMOTO and Yoshizo TAKAHASHI

Dept. of Information Science and Intelligent Systems
Faculty of Engineering, University of Tokushima

2-1 Minami-josanjima-cho,Tokushima 770,Japan

e-mail flavell@n30.is.tokushima-u.ac.jp

A scalable hierarchical interconnection network called MANDALA, suitable for the implementation of a massively parallel multicomputer system is proposed. The network is constructed by recursive, complete connection of processing elements (PEs). The PEs grouped are into clusters of size C and the network is formed by interconnecting L levels of these clusters. This yields a network with $N = C^L$ nodes, fixed degree, and diameter$D(L) = 2^L$-1. Under uniform routing conditions the average communications distance of the network, which is a function of the cluster size, is $O(N^{1/log_2C})$ while the maximum message density at the L-1 level interface nodes is $O(N^2)$. The communications nodes in the network are homogeneous, which aids in the easily scalable nature of the network . Simple algorithms for routing, broadcasting and multicasting are demonstrated and routing in the presence of faults is also discussed.

MANDALA: An Interconnection Network for a Massively Parallel Computer

Andrew FLAVELL, Takuya KANOH, Shigenori FUJIMOTO and Yoshizo TAKAHASHI

Dept. of Information Science and Intelligent Systems
Faculty of Engineering
University of Tokushima

2-1 Minami-josanjima-cho,Tokushima 770,Japan

e-mail flavell@n30.is.tokushima-u.ac.jp

ABSTRACT

A scalable hierarchical interconnection network called MANDALA, suitable for the implementation of a massively parallel multicomputer system is proposed. The network is constructed by recursive, complete connection of processing elements (PEs). The PEs grouped are into clusters of size C and the network is formed by interconnecting L levels of these clusters. This yields a network with $N = C^L$ nodes, fixed degree, and diameter $D(L) = 2^L\text{-}1$. Under uniform routing conditions the average communications distance of the network, which is a function of the cluster size, is $O(N^{1/log_2 C})$ while the maximum message density at the $L\text{-}1$ level interface nodes is $O(N^2)$. The communications nodes in the network are homogeneous, which aids in the easily scalable nature of the network . Simple algorithms for routing, broadcasting and multicasting are demonstrated and routing in the presence of faults is also discussed.

Keywords: multicomputer, interconnection network, routing, broadcasting, fault tolerance, scalable.

1. INTRODUCTION

The continued rapid advances being made in the field of VLSI technology have recently made possible, the commercial production of microprocessors containing approximately 2.5 million transistors[1]. The question of how the available silicon can best be put to use continues to provide an active avenue for research.

One possible option is the implementation of PEs for an multiple-instruction-multiple-data (MIMD) computer. MIMD systems can be divided into two groups, shared memory systems (multiprocessors) and distributed memory systems (multicomputers), both of which have advantages and disadvantages. Shared memory systems have a global shared memory that each processor can access and this provides a simple means of sharing both data and code. However, as each processor usually maintains it's own cache of frequently used data, the maintenance of cache coherency between processors can become a complex issue as the number of processors is increased[2]. Distributed memory systems are constructed by the interconnection of independent processing nodes, each of which has it's own local memory and an interface into an interconnection network (IN). The sharing of data and code between processors is achieved by explicit message passing, and thus the interconnection network can often restrict the overall performance of the system. It is, however, generally considered that multicomputer networks are more suitable for the implementation of systems containing hundreds or thousands of processors[3]. A large number of IN topologies have been proposed and implemented. These include the hypercube, torus, mesh, cube-connected cycle (CCC), and binary tree. However, these networks have a number of disadvantages, especially as the network size tends towards that required for massively parallel computing.

In this paper we introduce the MANDALA interconnection network. MANDALA is a scalable network suitable for the implementation of a massively parallel computer, which is realized through the interconnection of completely connected processor clusters. The cluster structure provides the maximum benefit for localized communications (intra-cluster communications), while providing an efficient communications path between distant clusters (inter-cluster communications).

A number of important issues must be addressed in the design of a network suitable for massively parallel computing. The average inter-node distance of the network should be kept as low as possible, and the degree of each node should be fixed. The routing algorithm should be simple, and there must be an effective broadcast-multicast mechanism. The network must remain operational, and should offer graceful degradation in performance in the presence of faults. Finally, the components of the IN should be homogeneous so that the overall system design is simplified, and so that advantage can be taken of the inexpensive replication of components[4].

2. NETWORK TOPOLOGY

The implementation of a massively parallel computer by interconnecting several thousand processors requires that the processors are grouped into clusters. These processing clusters (PCs) are then interconnected hierarchically as shown in Fig. 1.

The structures of the upper interconnection network (UIN) and lower interconnection network (LIN) may be identical, as in

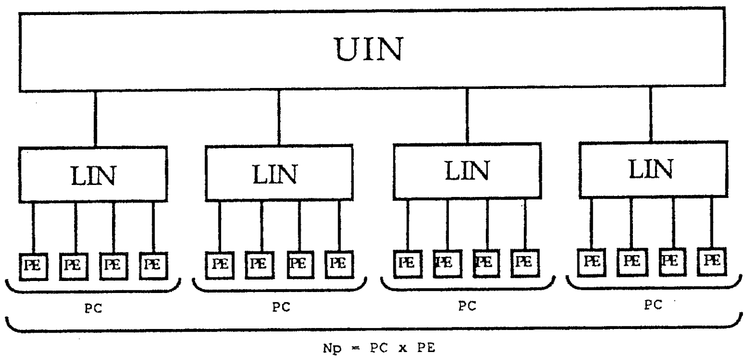

Figure 1: Two Level Hierarchical IN.

MANDALA, or different, as in HYPERNET[4]. A processor address *(ccccpppp)* is allocated such that the higher bits *(cccc)* represent the cluster number, while the lower bits *(pppp)* identify the processor within the cluster.

A number of authors have investigated the advantages of implementing parallel computers using hierarchical interconnection networks (HINs), but these networks have employed existing tree, ring and hypercube structures[4][6][7]. Our approach, which is similar to that taken in FIN-1 of Toyohashi Univ. of Technology[5], has been to develop a HIN by recursive use of a complete connection. However, unlike FIN-1, MANDALA is being developed as architecture suitable for massively parallel computing. Let C represent the number of PEs in a cluster. The C PEs are interconnected by a complete connection to form the first level in an L level hierarchy. C level 1 clusters are then interconnected to form a level 2 cluster. The number of PEs in a system is therefore given by:

$$N = C^L \qquad (1)$$

or, with respect to L:

$$L = \frac{log_2 N}{log_2 C} \qquad (2)$$

Fig. 2 illustrates a MANDALA network for $C=4$ and $L=3$. Each node in the network represents a communications processor to which a PE is attached. The number of processors in the network can be increased by either increasing the number of nodes per cluster or increasing the number of levels in the hierarchy. As the degree of each node is fixed in a practical system, the system would be scaled by increasing the number of levels. If required, a single node may be added to the system, providing that the rules for addressing an port connection are followed.

The node address in an L level network is represented by an L digit C-*ary* number n as:

$$(n_L n_{L-1} \ldots n_i \ldots n_1 n_0)$$

Port j of node $n_L, n_{L-1}, \ldots, n_1, i$ is connected to port i of node $n_L, n_{L-1}, \ldots, n_1, j$ when $j \neq i$. Port i of node i is reserved for connection to higher levels.

3. NETWORK CHARACTERISTICS

Some of the more important static evaluative measures of a network are its degree, diameter and average distance.[8] As a networks degree and average distance are inter-related, a number of authors have also suggested the use of normalized average distance[6][9] to provide a better measure of the latency of a link.

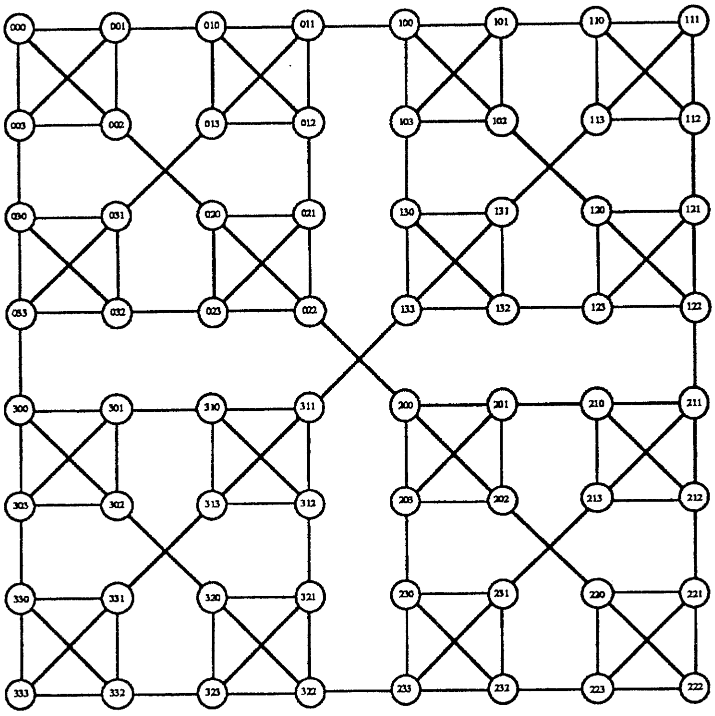

Figure 2: MANDALA Network with 64 PEs (C=4,L=3).

The average communication distance and the network diameter in MANDALA are denotedby $d(L)$ and $D(L)$ respectively. The network diameter is given by:

$$D(1) = 1, D(2) = 2D(1)+1, \ldots, D(L) = 2D(L-1)+1$$

therefore:

$$D(L) = 2^{L}-1 \qquad (3)$$

or, as a function of N, $D'(N)$ is given by:

$$D'(N) = N^{1/\log_2 C} - 1 \qquad (4)$$

3.1 Routing & Average Distance.

When a message with the destination address $d_L d_{L-1} \ldots d_i \ldots d_1 d_0$ arrives at a node whose address is $n_L n_{L-1} \ldots n_i \ldots n_1 n_0$, the digits of the two addresses are compared from most significant digit to least significant digit. If the two addresses are different, then the most significant digit of the destination address where the addresses differ, is the port number that the message should be output to. If the addresses are the same, then the message has reached the destination and can be forwarded to the PE.

A formal definition of this routing is presented in the function *Rsim()*. This function returns the output port for the message if the addresses differ or *DEST* if the message has reached it's destination.

```
Rsim(d,n)
char  d[],n[];
{
  level = L
  while(d[level]!=n[level]  &  level >0)  level--;
      if(level>0)  return  (d[level]);
  else return(DEST)
}
```

We have noted that this routing scheme does not necessarily result in the shortest possible path between two nodes. However, investigation has shown that although it is possible to reduce the number of communications nodes that some messages traverse, the high overhead involved in the route calculation negates any benefit obtained[12].

The average communications distance is now investigated. The average distance from any source node S to any destination node T , denoted $s(L)$, is found by first calculating the average distance from any of the interface nodes to T. There are C^L destination nodes (allowing $S = T$), of which C^{L-1} are in the same level L-1 cluster as the interface node. The remaining $(C^L$-$C^{L-1})$ nodes lie outside of the level L-1 cluster of S. When T belongs to the same level L-1 cluster as S, the average distance is $s(L$-$1)$. When T belongs to a different level L-1 cluster, the average distance is the sum of the diameter of the L-1 cluster and $s(L$-$1) + 1$.

Thus $s(L)$ is,

$$s(L) = \frac{(2^L\text{-}1)(C\text{-}1)}{C} \tag{5}$$

The average distance in the network can now be found. There are C^L-1 communications paths in total, of which C^{L-1}-1 are in the same level L-1 cluster as S. The remaining C^L-C^{L-1} extend into the other L-1 clusters, the average distance of these being $2s(L$-$1)+1$.

The average distance is therefore found to be:

$$d(L) = \frac{(2^L C^L\text{-}1)2(C\text{-}1)^2}{C^{L-1}(2C\text{-}1)C} - \frac{C\text{-}2}{C} \tag{6}$$

or as a function of N, denoted by $d'(N)$,

$$d'(N) = \frac{N^{1/log_2C}(N\text{-}1)2(C\text{-}1)^2}{(N\text{-}1)C(2C\text{-}1)} - \frac{C - 2}{C} \tag{7}$$

This is of the order,

$$d'(N) = O(N^{1/log_2C}) \tag{8}$$

3.2 Communication Load

If the communications paths terminating at a node are excluded, then the communication load of the nodes within MANDALA is dependent on the number of different communications paths passing through the node. As only the paths between nodes in different clusters of the same level pass through the interface node of this level, the communications load is the product of the number of nodes in a cluster and that of those in equal or higher levels. The communications load of the interface node at level i is denoted $r(i,L)$ and is calculated as follows:

$$\begin{aligned} r(L,L) &= 0 \\ r(L\text{-}1,L) &= (C^{L-1} - 1)C^{L-1} \\ r(L\text{-}2,L) &= (C^{L-2} - 1)(C^{L-2}+C^{L-1)} \\ &\vdots \\ r(i,L) &= (C^i - 1)(C^i+C^{i+1}+\ldots+C^{L-1}) \\ &= \frac{(C^L - C^i)(C^i - 1)}{(C\text{-}1)} \end{aligned} \tag{9}$$

4. TOPOLOGY COMPARISON

The static characteristics of a number of contemporary networks are investigated to evaluate the relative merits of the MANDALA network. This evaluation includes the link cost, which is represented by the degree of the link, average internode distance and maximum message density under uniform routing, the fault tolerance of the network, and the scalability of the network. The networks that are included in the comparison are hypercube, torus and mesh and binary tree. The results of the comparison are presented in Table 1.

As can be seen in Table 1, the MANDALA networks both exhibit good fault tolerance and scalability. The MANDALA network with a cluster size of 8 also exhibits lower average distance than the torus and mesh networks. Although the hypercube has the lowest average distance, it's degree varies with the dimension of the network (k) and it is not easily scaled.

Table 1: Comparison of Network Characteristics

	Average Distance	Degree	Normalized Average Distance	Scalability	Maximum Message density	Fault Tolerance
Hypercube	$0.5\log_2 N$	k	$k(0.5\log_2 N)$	POOR	$\frac{(N-1)(N\log_2 N-1)}{4}$	GOOD
Torus	$\frac{\sqrt{N}}{2}$	4	$2\sqrt{N}$	GOOD	$\frac{\sqrt{N^3}}{4}$	GOOD
Mesh	$\frac{2\sqrt{N}}{3}$	4	$\frac{8\sqrt{N}}{3}$	EXCELLENT	$\frac{\sqrt{N^3}}{2}$	GOOD
Binary Tree	$2\log_2 N$	3	$6\log_2 N$	EXCELLENT	$\frac{5N^2}{16}$	POOR
MANDALA (C = 4)	$\sqrt{N}$	4	$4\sqrt{N}$	EXCELLENT	$\frac{N(N-4)}{16}$	GOOD
MANDALA (C = 8)	$\sqrt[3]{N}$	8	$8\sqrt[3]{N}$	EXCELLENT	$\frac{N(N-8)}{64}$	GOOD

5. BROADCAST - MULTICAST

Many parallel computing applications require an efficient mechanism for the broadcasting of messages from one node to all of the other nodes in a system or multicasting to a subset thereof. These operations are used in matrix multiplication, LU-factorization, the implementation of parallel Prolog[10], and in problem solution by competing processors[11]. In this section we present two simple algorithms that implement both the broadcasting and multicasting operations. If the maximum level that the message is to be broadcast-multicast to is defined as *level,* then given an L level network a message is a *broadcast if level* $= L$. It follows then, that given an L level network, a message is a *multicast if level* $< L$. The address of a node has been defined as the L digit *C-ary* number, $n_L, n_{L-1}, \ldots, n_1, n_0$. Each node has C ports, excluding the connection to it's own PE, numbered, *port(0), port(1),.....port(C)*. *Port(x),* is defined as having a level 1 connection *(IntPort)* if $x \neq n_0$. Similarly, *port(x)* is defined as the external connection of a node *(ExtPort) if* $x = n_0$ .The node initiating the broadcast/multicast simultaneously writes the messages to all of the nodes within the same level 1 cluster, and then, writes the message to its external port connection. This is illustrated in the function *InitBroadcast()*.

```
InitBroadcast(Intports,ExtPort,level,extconn,Mess)
int IntPorts[C-1],ExtPort, level,extconn;
char Mess[PacketSize];
{
  SimWrite(Mess,level,IntPorts);
  extconn=PortConn(MyNode);
  if(exists(ExtPort)&&(extconn <= level){
       Write(Mess,level,ExtPort);
  }
}
```

The three arguments of *SimWrite() and Write()* are, respectively, the message to be propagated, the maximum level that the message may be passed, and the port/s that the message is to be sent to. The function *PortConn()* returns the level of connection of the external port of the node. This is determined from the address of the node. The external port of a node whose address is $n_L, n_{L-1}, \ldots n_0$ will interconnect levels of distance L_s+1, where L_s is the number consecutive digits that are identical from n_0 to n_L. A formal description of the action of propagating the message is presented in the function *PassBroadcast()*.

With reference to *PassBroadcast()* the message is propagated as follows. If the incident message arrived at a node via

```
PassBroadcast(IntPorts,MyNode,SrceNode,ExtPort,LevDiff,ExtNode,Mess,InpPort)
char Mess[PacketSize];
int  Intports[C-1],MyNode[L],SrceNode[L], ExtPort, LevDiff, ExtNode[L],InpPort;
{
  if(!IntPort) SimWrite(Mess,level,Intports);
  else{
      LevDiff = Difference(MyNode,SrceNode,L);
      extconn=PortConn(MyNode);
      if(extconn>LevDiff && extconn <= level) Write(Mess,level,Extport);
      else{
         ExtNode=NodeCalc(MyNode,L);
         if(ExtNode[L]==InpPort) Write(Mess,level,Extport);
         }
    }
}
```

ExtPort then the message is simultaneously written to all of the nodes within the same level 1 cluster. If however, the incoming message arrived at a node via *IntPort*, then there are two instances which, if valid, will allow the node to propagate the message further. If the external port of the node provides an interconnection of level L_i and the difference in levels between the source node and the current node is D_{sn}, then the node will propagate the message *if* $L_i > D_{sn}$ *&&* L_i *<= level*. The function *LevDiff()* returns an integer that is the difference between the current node and the source node, while *PortConn()* has the same function as discussed previously, and the function *NodeCalc()* returns the address of the node connected to the current node's external port. The other instance where a node may pass the message is detailed as follows. In any sub-cluster, the retransmission of the broadcast message may be viewed as propagating outwards from the interface node of that cluster along a number of verticies. Each time that a *Write()* operation takes place from within the *PassBroadcast()* function, the message is written to input port i from output port j. For each of the verticies, i and j remain constant. When this *Write()* operation is executed one further condition must be met to ensure that the message is not transmitted to a node that forms part of a neighboring cluster or sub-cluster. This can be viewed graphically as ensuring that the direction of the verticies does not alter when the message is written to the external port of the nodes. Careful examination of this graphical solution yields the following rule. In this instance, *Write()* can only be executed *if ExtNode*(n_0) *= InpPort*.

6. ROUTING IN THE PRESENCE OF FAULTS

As the size the interconnection network is increased, the mean time between failure (MTBF) of the system as a whole decreases. Therefore, the interconnection network for a massively parallel computer must provide a degree of fault tolerance. The distribution of interconnections between clusters provides a large number alternate routes should re-routing of a message be necessary. Should a fault occur in a communications node, or the message traffic be such that a message packet effectively becomes blocked, these paths can then be utilized. A message packet will be re-routed around a faulty or blocked node, using the shortest alternative path. This is determined while the message packet is queued, waiting for the appropriate output port. Should the desired output port become available, the packet will be passed to that port and the alternate path information is ignored. If however, the packet remains blocked, the message can be re-directed via the alternate path.

7. CONCLUSIONS

In this paper we have presented a discussion of the MANDALA interconnection network. This network is constructed by the recursive interconnection of completely connected PEs. The distribution of the interconnection between the *(L-1)* clusters provides a good degree of fault tolerance. The degree of each node is fixed and as few as one processor can be added to the network if it is to be scaled. All of the communications nodes of the network are homogeneous, allowing the replication of a single design for all of the nodes in the system. We have demonstrated very simple routing and broadcast-multicast algorithms, which will aid in the construction of communications nodes with low switching latency. We believe therefore, that the MANDALA topology is a good candidate for the implementation of a massively parallel computer.

References

[1] Myers W., Contributing Editor, *Caltech Dedicates Worlds Most Powerful Supercomputer,* IEEE Comp. Magazine, July 1991, pp. 96-97

[2] Duncan R., *A Survey of Parallel Computer Architectures,* IEEE Comp. Magazine, Feb. 1990, pp. 5-16

[3] Seitz C. L., *The Cosmic Cube,* Commun. ACM, Vol. 28, No. 1, pp.22-33 (1985)

[4] Hwang K., Gosh J., *Hypernet: A Communication-Efficient Architecture for Constructing Massively Parallel Computers,* IEEE Trans. on Computers, Vol. 36, No. 12, pp.1450-1466 (1987)

[5] Sugaya M., et.al., *Some Parallel Algorithms on a Multiprocessor System with Fractal Geometry-Based Interconnection Network,* The Trans. of the Inst. of Electronics, Information and Communication Engineers, Vol. J73-D-I, No.11, pp.847-855 (1990) (in Japanese)

[6] Dandamundi S. P., Eager D. L., *Hierarchical Interconnection Network for Multicomputer Systems,* IEEE Trans. on Computers, Vol. 39, No. 6, pp. 786-797 (1990)

[7] Wu S. B., Liu M. T., *A Cluster Structure as an Interconnection Network for Large Multi-microcomputer Systems,* IEEE Trans. on Computers, Vol. C30, No.4, pp.254-264 (1981)

[8] Agrawal D. P., Virendra J. K., *Evaluating the Performance of Multicomputer Configurations,* IEEE Comp. Magazine, May 1986, pp.23-37

[9] Reid D. A., *The Performance of Multicomputer Interconnection Networks,* IEEE Comp. Magazine, June 1987, pp. 63-73

[10] Takahashi Y., Inoue K., Endo T., *Performance of Parallel Execution of Prolog Program by Goal Broadcasting on the Binary-Tree Parallel Computer,* Proc International Computer Symposium, Taipei, Vol. 1, pp 743-747, (1988)

[11] Takahashi Y., Sano M., *Parallel Computing with a Number of Competing Processors,* Parallel Computing '91, London 1991 (to be published)

[12] Fujimoto S., *Minimum Path Routing in MANDALA,* Technical Report of Parallel Computing Lab., University of Tokushima, August 1991