Report 77-31
Stanford -- KSL
Scientific DataLink

Some Issues in the Design of Large Multi-Microprocessor Networks.
Kjell G. Knutsen, Aug 1977

HPP-77-31

SOME ISSUES IN THE DESIGN OF LARGE MULTI-MICROPROCESSOR NETWORKS

Kjell G. Knutsen[1]
Computer Science Department
Stanford University
August, 1977

1. INTRODUCTION.

The low cost of microprocessors today, and the future trend in both cost and performance, makes large microprocessor-networks very interesting. A net of a thousand processors or more can be built using present techniques. However, there are several problems in utilizing such a horde of processors in a reasonably efficient way. There also seem to be restrictions to the kinds of applications that can be mapped on to such a system. One control mechanism that seems to be very useful, at least for some Artificial Intelligence type problems, is the CONTRACT NET, [Smith77].

In this paper we will first look at some of the desirable characteristics of a large multi-microprocessor net. Then we will describe several different organizations together with their advantages and problems. We are also discussing broadcasting in lattices using circuit switched minimum spanning trees.

2. CHARACTERISTICS

The main features of microprocessors are their performance/price and performance/size ratios. For many applications where a multi-microprocessor is an alternative, a single mini- or mainframe computer could have been used instead. In such a case, if cost is one of the reasons for using microprocessors, one must consider very carefully the cost of interconnecting these. In other applications, where the processors have to be geographically distributed, the communications network is an inherent part of the system, and higher communications cost can be justified.

2.1 Modularity and identical components

Microprocessors also offer an obvious potential for modularity and mass production. A processor node must at least include the following:

- communications interface
- processor
- memory

---

[1] This work was supported by the Advanced Research Projects Agency under contract DAHC 15-73-C-0435. Computer facilities were provided by the SUMEX-AIM facility at Stanford University under National Institutes of Health grant RR-00785. The author is supported by the Royal Norwegian Council for Scientific and Industrial Research and also by the Norwegian Department of Defense. Hector Garcia, Reid G. Smith and Gio Wiederhold have been a great help in preparing the manuscript.

The communications interface might drive several lines and channels, and might very well be processor controlled. In fact, a "microcomputer on a chip", like the Intel 8048, might be a good choice for control of the communications interface.

The communications interface is very tightly tailored to the communications technique(s) used. If one technique, e.g. radio, is the only one used in a network, it does not seem unreasonable to expect that the communications interface can be the same in all nodes, and supporting the same protocol(s). If this is not the case, e.g. if radio is used in some subnetwork while direct link connections are used elsewhere, a limited number of communications interface types must be included as system components.

For the processor and memory parts of the node we would also like as few different types as is practical. But we would like to be able to provide some processors with special hardware (multipliers, FFT-processors etc), and also to try out and incorporate new processors resulting from advances in technology. The latter is also true for memory. Furthermore the memory should be matched to the processor(s) it is serving.

2.2 Reliability and failsoftness.

There are several factors determining the reliability and failsoftness of a system:

- Physical organization and layout. The organization would have to provide for more than one communication path between any two nodes. No single processor failure should be fatal to the system. The very best would be if only a degradation of the system corresponding to the processing capacity of the failing processor, took place. This needs to be supported by the

- Control structure. We are not considering very centralized organizations or control mechanisms, but want to distribute control throughout the net. It also ought to be possible for another processor to take over the task(s) of a failing processor. We are not interested in fixed routing schemes, where the direction of a message is determined from a part of the address (as in most telephone networks). One solution for a packet switched network is to use dynamic routing tables. These can be regularily updated and have entries for all nodes. For 1000 nodes or more that will require large tables in each node. An alternative is to use fixed table length with a number of entries equal to for example, 10 % of the total number of nodes. The least recently used entry could then be deleted when a new entry is needed. In such a scheme, it will happen that a needed entry is not found in the routing table. It must therefore be possible to search for a node by name (number), and thereby update the routing tables.

- Hardware. The reliability of a system obviously depends heavily on the reliabilty of its hardware parts.

2.3 The CONTRACT NET mechanism.

For the control structure part of the reliability problem the CONTRACT NET mechanism seems to offer a good solution. In short, as a processor node is defining subtasks of its current task, these are either announced to the net in the form of contracts, or saved to be taken on by the node itself. Interested nodes, i.e. those not too busy, or perhaps, possessing special hardware or software requested, bid on the contracts. The contract is then awarded to the node offering the most attractive bid, and a two way link is established between the contract manager and the contractor. The contract is terminated with a report from the contractor to the contract manager, - or if the manager chooses to terminate.

This mechanism effectively distributes control between the contract managers. And the links make it possible for the manager to keep track of subtasks. If a contractor is not responding, the manager can reannounce a contract. Also, the contractor can decide whether or not a manager is responding.

2.4 Messages

In computer networks one might often want to broadcast a message,- i.e. to deliver the same message to all nodes in the net. Especially in large nets, an efficient way of broadcasting can be important for the overall performance. The most obvious way of broadcasting is by means of radio-transmission, but it is also possible to broadcast in lattices and other networks using direct links for interconnections.

We will use the following abbreviations for message types in the net:

BC - broadcasting (one to all other nodes)

GM - group message (one to some)

PP - point-to-point (one node to another)

For an implementation of the CONTRACT NET control mechanism (see 2.3), different types of information need to be transmitted,

Message	Type	Size	Frequency
Announcement	BC (or GM,PP)	small	medium
Bid	PP	small	often
Problem descr. (Award)	PP (or GM)	medium to large	medium
Results (Reports)	PP (or GM,BC)	application dependent	medium
Termination	PP (or GM,BC)	small	seldom to medium

The Announcement message describes the problem very shortly. It can be broadcasted or sent to a group only, or even sent to a single node, preferred because of properties already known. Each announcement will generate bids, the number depending on bidding and awarding policies and also on the state of the processors in the net. An award must give a complete problem description, and might include programs needed. The size of the results will depend on the application, and also the kind of subtask it is an answer to. The termination message will be used mainly to terminate execution when a sufficient number of solutions to a problem have been found.

An underlying assumption for the rest of the paper is therefore that most of the traffic is between two nodes (PP), or concerning only a limited group (GM). However, broadcasting is a very important feature for the Announcement message.

2.5 Parameters

Throughout the paper we will use abbreviations for the following parameters (reasonable ranges are given to the right):

Total channel bandwidth	W	< 10**8 ch/sec (approx 1 Gb/sec)
Coupling	C	10**-2 - 10**-7 ch/8080-instr
Processor speed	S	10**6 - 10**7 instr/sec
Processor power	P	1 - 100 8080-instr/instr
Total no of processors	N	10**3 - 10**7

In this context, the coupling is the average number of characters received per 8080-instruction executed in a processor node. As can be seen from the range of this parameter, C, we are assuming rather loosely coupled nodes, i.e. a processor spends most of its time "working", and much less communicating with the other processors. C is a very critical parameter, as will be shown, and determines to a large extent he maximum number of nodes that can be used in a system.

3. ORGANIZATIONS.

3.1 Radio

We will consider both using a common channel, and splitting up the channel into several frequency bands.

Radio transmission using a common channel is logically the simplest way of organizing a net of nodes. It is very attractive in that it offers direct connections between any two nodes and also makes broadcasting very simple and efficient. One can tolerate some geographical distribution and reconfigure the net very easily. Neither will there be any routing problems. For some

applications, like a distributed sensor-net where sonarbuoys are dropped from airplanes, this can be the only viable solution. For military systems, spread spectrum modulation can be used to hide any ongoing communication.

The main limitation to the use of radio on a common channel as the only communication technique in a large system, is bandwidth. The total required bandwidth will be proportional to the number of processors, N. If we do not take into account the time used by a processor for communication, thereby assuming loosely coupled nodes, we have

(3.1.i) W = N * P * S * C ch/sec

where the parameters given in 2.5 have been used. The bandwidth is also proportional to the processor performance (P*S), and the coupling factor. The faster a processor executes a task, the sooner it is ready for a new task (which requires communication). The inclusion of C is also rather obvious, since the more characters are received per instruction executed, the more bandwidth will be needed.

For a typical 8080 system, we have:

P = 1 8080-instr/instr

S = 5*10**5 instr/sec

giving

(3.1.ii) W = N * 5*10**5 * C ch/sec

In figure 3.1 we have drawn lines for different values of the required bandwidth W (in ch/sec), varying N and C.

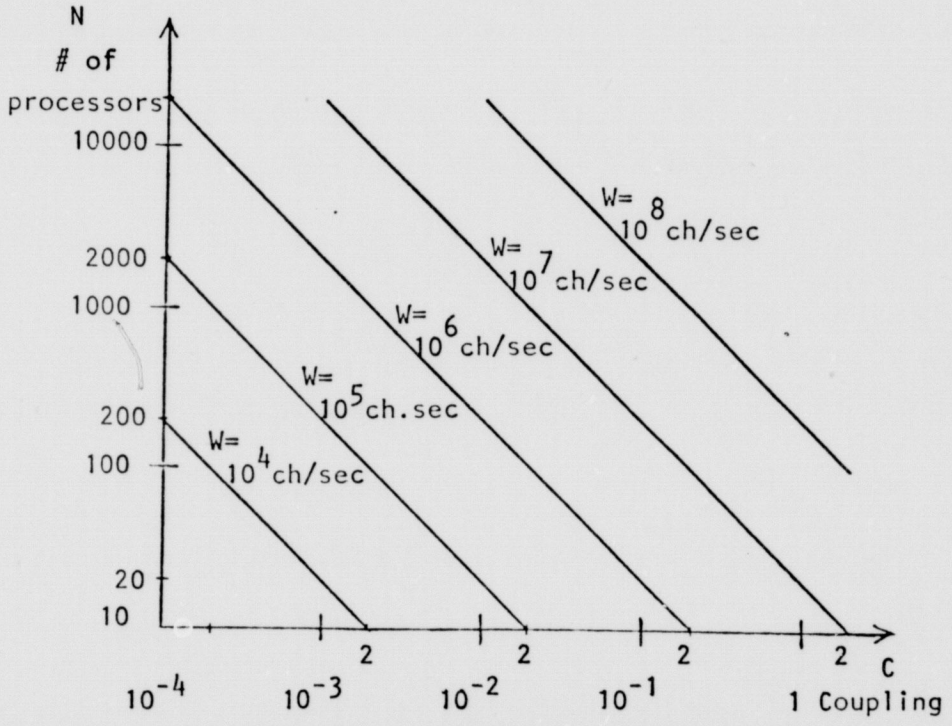


Fig. 3.1. Required band width, W, for different values of N, # of processors, and the coupling, C.

As we can see, a bandwidth limit of 10**8 ch/sec (which is very high in such systems), makes possible both a large number of processors (> 1000), and rather closely coupled processors (C=.1). However, what is very likely to happen in the next few years, is that microprocessor performance is going up, and it is not unlikely that we will see a P*S 10 to 100 times that of an 8080 in the beginning of the 1980's. A P*S equal to 5*10**7 changes W to

(3.1.iii) W = N * 5*10**7 * C ch/sec

which will move the W=10**8 ch/sec line down to where the 10**6 ch/sec line is in figure 3.1. The lines for other values of W will be moved in the same way.

Also, we would like to reduce the bandwidth, to avoid expensive buffering circuitry in the communications interface. If in addition we want to allow for more than 1000 processor nodes in a system, there seems to be no other way than to look for other solutions of the communications problem.

To reduce the bandwidth and thereby the cost of the communications interfaces, one could split up the available bandwidth, so that not all processors had access to all of the frequency bands. One solution would be to have one frequency band which was used by all nodes for broadcasting and then several others which were used by groups of nodes. One could let such a group configuration be dynamic, by letting each node decide which frequency band to use according to the current task. It is, however, questionable if the cost saved by using a lower data rate into the buffers, will not be counteracted by the higher complexity. Neither will there be any bandwidth gain from such a splitting, rather a lower efficiency in bandwidth utilization. This is so because a large number of sources, using a high bandwidth channel, will tend to use the channel more evenly due to randomness of their demand, than would be possible for smaller groups. Also, a group's traffic interests will not be as independent and random as that of the total system.

3.2 Closed Circuit Radio.

Closed circuit radio has most of the same limitations as radio. Some advantages are

- no interference from external sources

- many parallel circuits can use the same frequency bands

The latter point can be utilized in at least two ways. One can let all processors be connected to all cables, thus increasing the available bandwidth, but only proportionally to the number of cables, see fig. 3.2 a. Another approach, similar to the one used in sec. 3.1, for radio, would be to connect groups of nodes to different cables, and have one common cable for broadcasting messages concerning all nodes, (fig. 3.2 b). This could be done dynamically as groups and topology changed, but that would mean added complexity and higher cost.

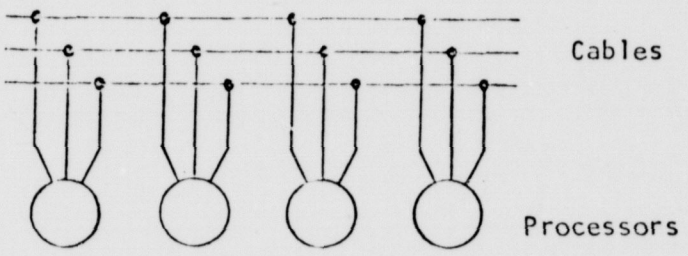


Fig. 3.2.a Closed Circuit Radio on many parallel cables.

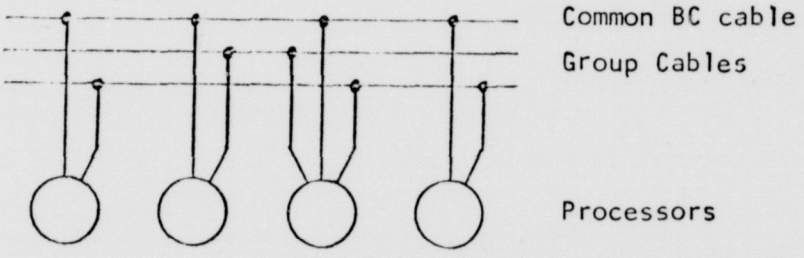


Fig. 3.2.b Closed Circuit Radio using one common cable for broadcasting and a number of others for internal group communication.

The main restriction compared to radio, is that all nodes have to be connected to one or more cables, thus making reconfiguration and movement of the whole net more difficult.

One example where this technology is being used, is in the Ethernet [Metcalfe76]. The machines used there are minis or larger, and are therefore not quite comperable in cost to microcomputers. Further, the number of machines connected to one cable, the Ether, is at most in the hundreds. And the machines are not supposed to work together all on the same task, which implies a very loose coupling. To increase the number of nodes in a system, several ethers can be connected by means of Gateways, which then take care of buffering and routing.

3.3 Bus

A bus is different from closed circuit radio in that it has a central controller (fig 3.3)

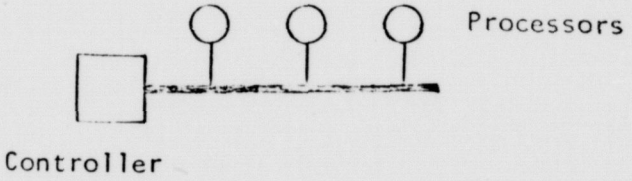


Fig. 3.3 Bus.

This makes a system consisting of only one bus, very vulnerable to failures in the bus controller. Also, the processing capcity of the controller is limited. Therefore, for large bus systems, several busses should be connected, for example as in fig 3.4.

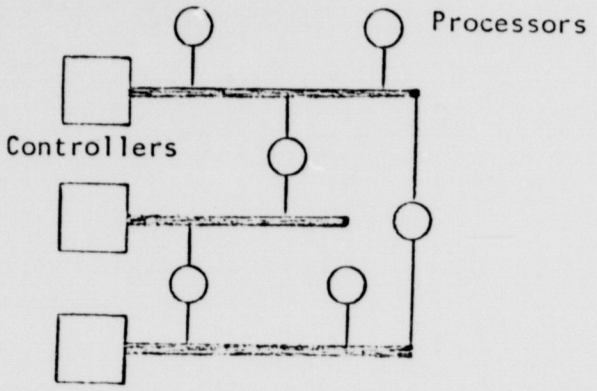


Fig. 3.4 Bus system.

If a central control is introduced in a system of many busses, this should be taking care of error detection and performance evaluation, rather than controlling and scheduling execution directly. In this way, a distributed control (among the bus controllers and the individual processor nodes) can still give very high reliability without duplication of units and other expensive precautions.

One can also think of a system where all communication between different busses goes through the bus controller (fig 3.5)

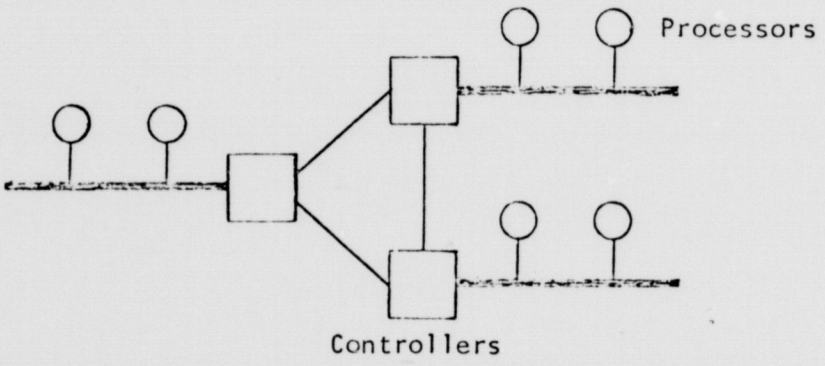


Fig. 3.5 Bus system where all interbus communication goes through the controllers.

The communications medium between the bus-controllers might very well be radio or closed circuit radio instead of point-to-point connections as shown in the figure. If one or more levels of busses are used in a hierarchical organization, however, then the reliability can never be better than that of the controller at the highest level.

3.4 Direct link connections.

In this category we will include regular networks such as lattices and rings as well as more irregular approaches.

3.4.1 Lattices

Lattices can be of different orders (dimensions) as shown in figure 3.6:

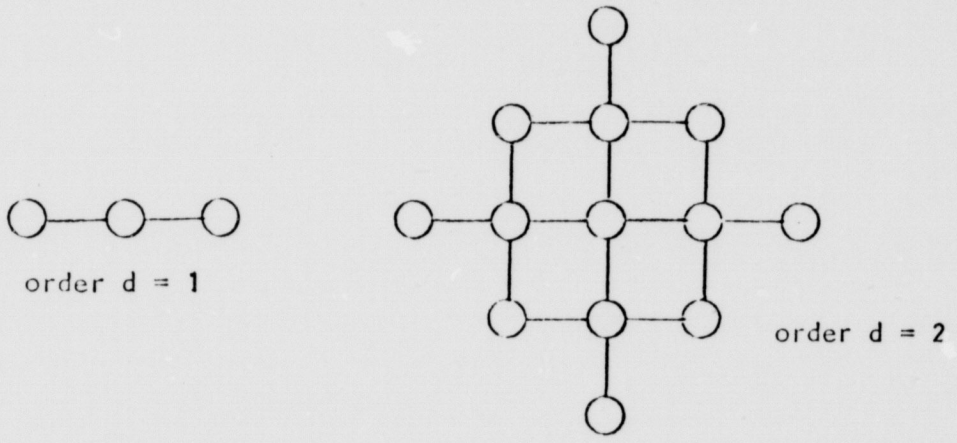


Fig. 3.6 Lattices

In a strictly regular lattice, all of the nodes will have 2*d neighbors, where d is the order of the lattice. The nodes at the edges of a lattice as shown in figure 3.6 all have the same distance from the center node. These nodes can be interconnected in a regular way forming a strictly regular lattice. In such a lattice, every node will have its most distant node at the same distance. Since we are considering using in the order of thousands of processor nodes, we prefer regular ways of connecting them. In that respect, lattices offer a very good solution. Lattices of order 2,3 and 4 are especially interesting:

- order 2. One plane geometry, easy to remove and exchange nodes
- order 3. More compact, but still the topology is easy to layout and to think of.
- order 4. Very compact, which means more nodes in shorter distances (fewer links) and higher bandwidth available for a given number of nodes. Also, a number of lines that is a power of two, would for some implementations of the communications interface, be optimal (memory sizes etc.). However, the physical layout and cabling is not as easy as for orders 2 and 3.

We have shown in an appendix that the distance (# of hops or links) between any two nodes in a regular lattice of order d is approximately

R = d'th root of N'

If we choose one of the two nodes as center node, then N' is the number of nodes having a distance from this center node less or equal to R. If N' is the total number of nodes in regular lattice, then we can call R the radius of the lattice. If the lattice is stricly regular, i.e. the edge nodes are connected in a regular way, then the longest distance between any two nodes is R.

In a regular lattice of N nodes and order d as described above, the nodes at the edges can be connected to other edge nodes, forming a regular interconnection network. The number of links in such a net will be

$$L = 2 * d * N$$

For PP messages, the worst case is when R links are used for a connection. If we assume that most of the messages are of type PP and that each link has an available bandwidth of LBW (ch/sec), then the available bandwidth is

$$ABW = LBW * 2 * d / R = LBW * 2 * d / d'th\text{-}root(N)$$

This is of course a very rough approximation, which does not take into account the routing problems when the shortest path is busy. However, there are usually a lot of paths having a length less than or equal to R.

If we compare with radio communications (sec. 3.1) using

C - .2 ch/8080-instr

N - 1000 processor nodes

P*S - 5*10**5 8080-instr/sec

then the required communication bandwidth is

N * P*S * C ch/sec = 10**8 ch/sec

For a lattice of order 3 we must have (if the available bandwidth is to be greater than the required)

$$LBW * 2 * d * N / d'th\text{-}root(N) > 10**8 \text{ (ch/sec)}$$

which means that

$$LBW > 1.7*10**5 \text{ ch/sec}$$

This is not at all difficult to meet. In fact, we can allow for a larger number of more powerful processors in the net.

While the required bandwidth when radio is used for communication, increases proportionally to N, the bandwidth requirements is only increasing as the d'th-root of N in lattices.

3.4.2 Rings

In a ring network a message is passed from one communication interface to its neighbor, either in one or both directions, until the destination(s) are reached (fig 3.7).

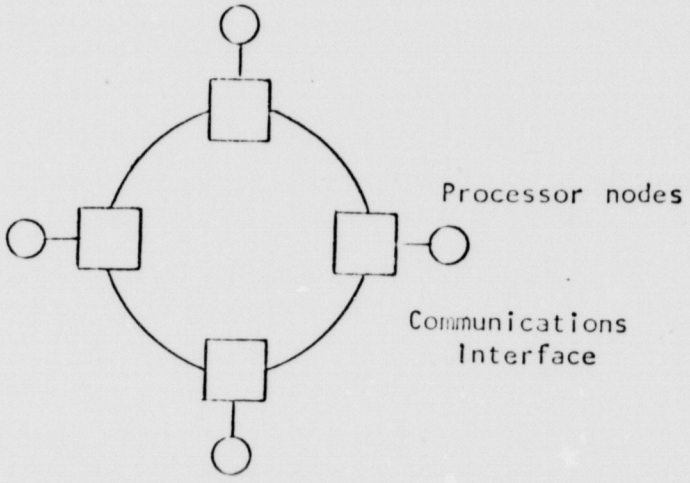


Fig. 3.7 Ring Network

An example of such a network is the Distributed Computer System (DCS) developed at UC Irvine [Farber73].

It is, however, very hard to think of one single ring having a thousand nodes or more. Both delays and available communications bandwidth will be critical parameters. Also such a system would be very vulnerable to breakdowns, both among the communication lines and in the communications interfaces. With any one connection down, there would be no more ring, and

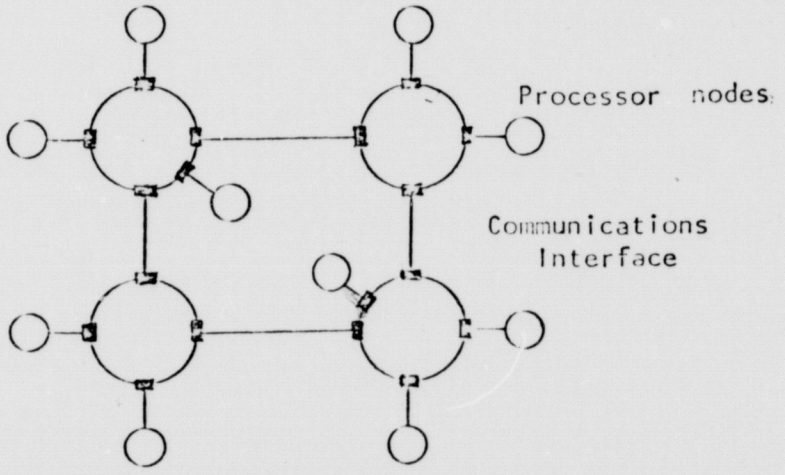


Fig. 3.8 Multi-Ring Network

two connections broken would split the ring into two separate networks. This last problem can be handled by means of redundant rings (as is done at UCI), but this necessarily makes the system much more complex. Therefore, if the concept of rings should be used, many smaller rings have to be connected together, e.g. like in fig 3.8.

Such a network would have most of the same characteristics as a lattice, but with more complex analysis, and is therefore not treated in more detail here. Also, for a large number of processors, the lattice structure is more homogeneous, and should therefore be easier to layout, operate and maintain.

3.4.3 Other

In practice, irregular networks will be more likely. Reasons for this can be geographical restrictions, alterations of the network or communication links and processors being out of order or maintained. In a lattice of 1000 processors or more, a few out of order, scattered around in the net, would not degrade performance significantly. In cases where geographical distribution is inherent in the system, one might use clusters of processor nodes where processing capacity is needed, and then connect these with a reasonable number of links. In that way, one can get very good reliability and have an easy reconfigurable network consisting of very few different modules.

4. PACKET SWITCHING AND CIRCUIT SWITCHING.

In a store-and-forward pecket switched network with direct link connections, each packet is held a short time in each switching node for examination. Then the packet is forwarded in the right direction, using routing tables, until the destination is reached. The ARPANET is an example of this [Kahn76].

The telephone system is an example of a circuit switched network. Here a connection is set up between source and destination, and as long as this connection is kept, it is used exclusively by either source or destination.

An important characteristic of the telephone network is that, on the average, the setup time for a connection is much shorter than the time used for talking. Also, when the connection is used for human conversation, the channel is utilized most of the time.

For data traffic the characteristics can be very different. Between computers high bandwidth is usually required to give reasonable response, and the traffic can be very bursty. Keyboard input from terminals is very low bandwidth, but for the response from the computer, a much higher bandwidth is desired.

To conclude, circuit switching is better when the amount of information to be transmitted is evenly distributed over time, not bursty, and this state lasts for a long time compared to the setup time for the connection. On the other hand, packet switching favors small bursty messages, to be sent to a lot of different destinations within a short interval of time.

5. BROADCASTING IN DIRECT LINK NETWORKS.

Dalal [Dalal77], has given a very good overview of broadcasting in direct link networks, assuming store-and-forward packet switching. Besides giving routing algorithms and analyzing these, he also gives algorithms for construction of minimum spanning trees (MSTs) in packet switched direct link networks. A MST is a subnetwork that includes all nodes but only a minimum number of links (edges) to interconnect these. The links that are in the MST are called "branches" of the MST, and are chosen so that the cost of broadcasting is minimized. This cost can range from infinite to zero. One of the MST algorithms has an adaptive version and is distributed.

This distributed and adaptive algorithm constructs a new minimum spanning tree from an old one . The algorithm starts in parallel in all nodes at the "leaves" (end nodes on a "branch"), of the old minimum spanning tree. All of these leaf nodes have one, and only one, branch connecting the node to the rest of the spanning tree. This branch is removed, and a new connection is made to its nearest neighbor, thus forming a "fragment" of the new MST. Thereby new nodes become leaves of the old MST, remove their branch to the old MST, connect to their nearest neighbors and form fragments, and so on. The fragments again will combine to form larger fragments, and in the end form a new MST. Each fragment has a "master" node. When fragments combine, the new fragment's master is unambiguosly agreed upon. So, when there is only one fragment left, the new MST, there will be only one master. This master node will "know" that the forming of a new MST is finished, and will broadcast the message "done", thus informing all nodes in the net that they can start using the new MST. The algorithm also allows for nodes and links to go down and reappear in the net, and also for new nodes to be introduced.

In a network using the CONTRACT NET mechanism, broadcasting is very central. Especially for announcements of contracts, but also for messages as "One solution found", when a given number (>1) of solutions are sought.

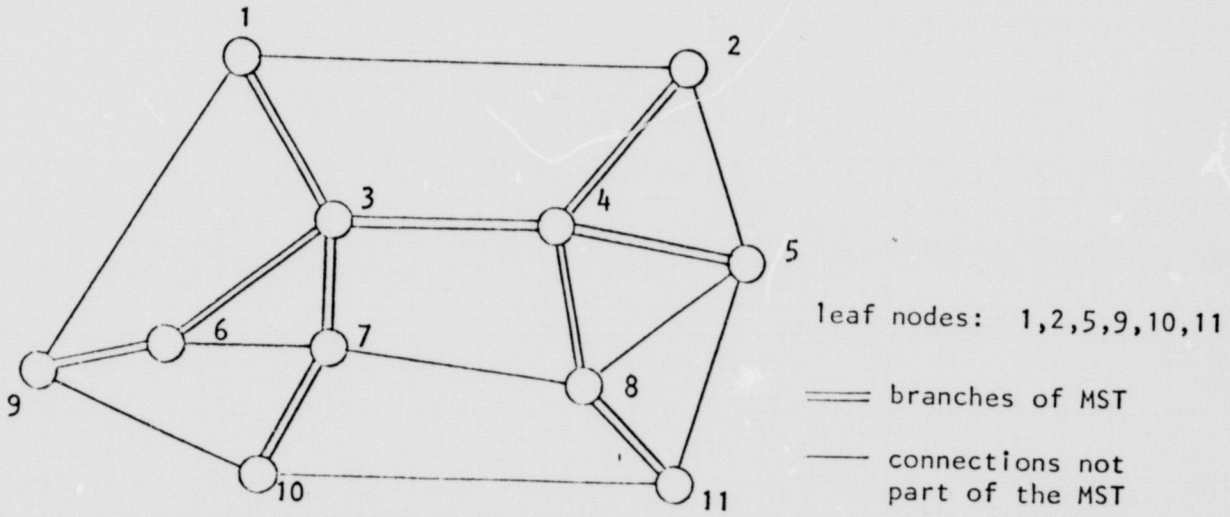


Fig. 5.1 A minimum spanning tree

Dalal developed the adaptive, distributed MST-algorithm for broadcasting in packet switched direct link networks. One of the disadvantages of the broadcasting by forwarding along a MST, is that this does not necessarily minimize propagation delay. Applying new electronic switching techniques, one can build digital switches (eg switching PCM formatted channels) where one source can be "simultaneously connected" to several destination channels [Drageset75]. By means of such switches, a MST can be set up utilizing electronic circuit switching, and thus connecting all nodes in a network together (see fig 5.1)

The advantages of using a circuit switched MST is

- Speed. Electronic circuit switching is much faster than packet switching once the connection has been set up.

- Processing requirements. There is no processing required in each switching node once a connection is set up. However, for extremely unstable nets, relatively much time can be spent on computing and setting up new MST connections.

The bandwidth requirements problem has two aspects

- The packets have to be longer in a packet switched network. (At least to distinguish BC messages from other kinds of messages. For group messages, if switched MST's are set up between the members, the savings can be more significant).

- The channel utilization might not be as good when one or more channnels are used exclusively for broadcast. That depends on how evenly distributed the broadcast traffic is, and how close to the actual demand it is possible to allocate bandwidth.

When a broadcasting message is being generated in a node, that node "knows" unambiguously on which lines (ie to which neighbors) to send the message. And all subsequent nodes getting the same message, "know" if and where to forward the message. For example, for node #6 in fig 5.1, one would have

a. If a BC message is generated in this node, send to 3 and 9.

b. If a BC message is coming from 3, send to 9, and vice versa.

One can also set up MST's for subnets formed by groups in a system, if there are messages to be broadcasted to all the members of the group. These MST's would have to allow for switching nodes being employed in switching and conveying the messages, without being a member of the group.

6. AN APPLICATION: A DISTRIBUTED SENSOR-NET.

In a distributed sensor-net there will be geographical distribution. Also, one might want to use many microcomputers instead of one or a few larger computers. The reasons for this can be many: reliability, reconfigurability, cost, homogeneity etc. If the net includes for example sonarbuoys dropped from airplanes, the only way to connect these is by means of radio. As an

example, such a system could have clusters of microcomputers on ships, airplanes or in installations on the ground, interconnected by cables in some sort of lattice to save radio bandwidth. These clusters could then be interconnected by means of cables, radio-links or radio (as would be the case with the sonarbuoys) to form a network (fig 6.1).

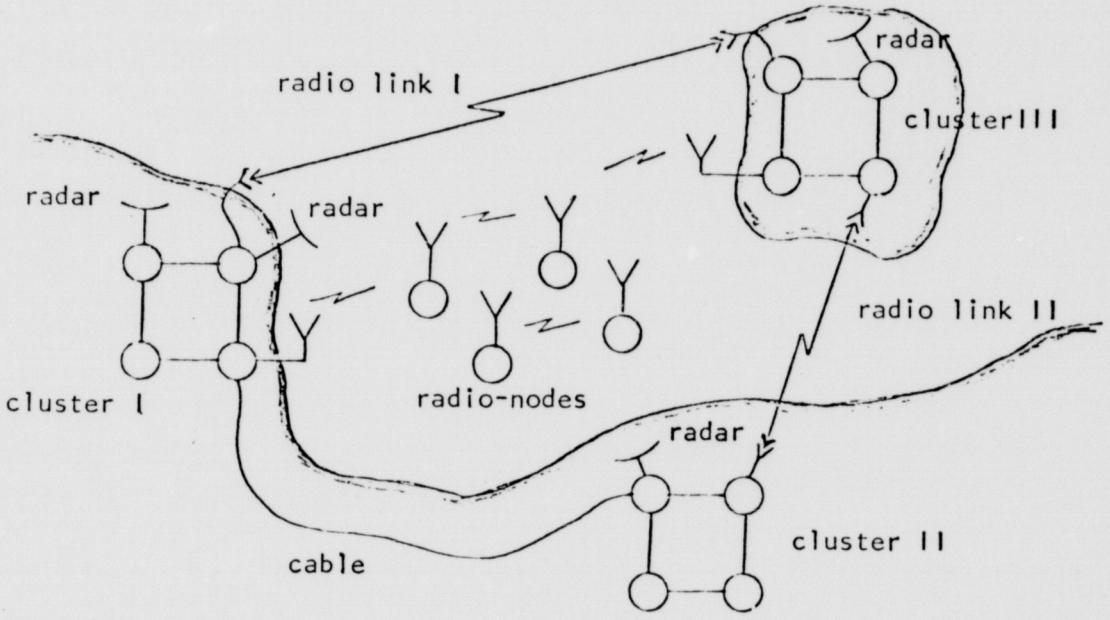


Fig. 6.1 Sensor - net example

We will call the nodes that have only an antenna for communication, radio-nodes (the sonarbuoys in our example).

Broadcasting can be accomplished in such a network in several different ways (eg. Hot Potato forwarding, Multi-Destination Addressing, Forwarding along a minimum spanning tree (MST) etc. [Dalal77 pp 110]).

If one wants to construct a MST, Dalal's algorithm (as was described in sec. 3.4.5) can be used, except in the radio-net used by the sonarbuoys. If these nodes too, are going to receive one and only one copy of broadcast messages, then each message must be transmitted by one and only one of the transmitting antennas (connected to clusters I and III in figure 3.10)

Dalal's adaptive, distributed algorithm is restricted to direct link networks. We will now sketch how this can be extended such that subnetworks using radio can be included in the MST. An assumption for the following is that all radio-nodes and antennas connected to the rest of the net can detect transmissions by all other sources. This is best described by a fully connected network. If the network is not fully connected, then the missing connections would have to be identified, and more complex schemes used.

When a new MST is to be constructed, the radio-nodes decide upon a master, (eg. the one with the highest identity number), and this node then represents all the radio-nodes. The master node needs to know the identity of the neighbor(s) in the old MST. This can easily be obtained by the master node by transmitting a message asking all link-nodes (ie those having both antennas and also link connections to other nodes) for the information. Another solution would be that all radio-nodes stored this information. If the radio-nodes were a leaf node of the old MST, then the master node would remove the branch to its neighbor (by sending a message to that effect), and "connect" to the nearest neighbor and form a fragment of the new MST. If the radio-nodes were not a leaf of the old MST, then the master would wait for one or more removal messages to make them a leaf. The rest of the algorithm is similar to the original one.

For zero cost of radio-communication, radio will be used for as much of the MST as possible. If the cost of radiocommunication is very high, then the radio will be used only to connect the radio-nodes and parts of the net inaccessible by other means.

In a conventional MST, only using direct link connections, every node decides if and where to forward a broadcast message. In a radio-network, all antennas will receive all messages, and therefore have to reject them if the antenna is not a part of the MST. If a circuit switched MST as desribed in sec.3.4.5, is used, the rejection can easily be done in hardware by disconnecting the broadcasting channel(s) on the antenna link.

7. CONCLUSION.

For small and loosely coupled networks, either radio or closed circuit radio yields a good solution to the communications problem. As the systems grow larger, with more and more powerful nodes, these solutions reach a bandwidth limit. To increase the available system bandwidth, the system must be partioned and some hierarchical or homogeneous interconnection method be employed. For reliability reasons we have been concentrating on homogeneous methods, of which regular lattices are examples. The regular structures are easy to layout, and nice for handling and control of processors in the thousands.

We have advocated using the CONTRACT NET as a control mechanism in the net. In this mechanism broadcasting is very central, and we have discussed how distributed algorithms can be used to form circuit switched minimum spanning trees, which have very desirable characteristics.

For some applications, as a distributed sensor-net, one solution of the communications problem is to use radiocommunication where cables or radio-links cannot be used. In places where processing capacity is needed (eg. onboard a ship), clusters of processors, organized as lattices, can be used to save radio bandwidth.

In this paper we have concentrated on organizations for loosely coupled multiprocessors and not dealt with control issues (eg. how to map problems on to a distributed environment) which is another very important aspect of distributed computing.

9. REFERENCES.

[Dalal77] Y.K. Dalal, "Broadcast Protocols in Packet Switched Computer Networks", Stanford Ph. D Th., Digital Systems Laboratories Technical report no 128, April 1977.

[Drageset75] O. Drageset, "Nodal Exchange - Switching Hardware", Norwegian Defense Research Establishment, TN - E - 686, 1975.

[Farber73] D.J. Farber et. al., "The Distributed Computing System", Proc. 7th. Ann. IEEE Computer Soc. International Conf. Feb. 1973.

[Kahn76] R. E. Kahn, "Techniques for Handling Stream Traffic via Packet Switching", Oral Presentation only, IEEE Compcon Spring, February 1976.

[Metcalfe76] R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks", CACM, vol 19, no 7, July 1976.

[Smith77] R. G. Smith, "The CONTRACT NET: A Formalism for the Control of Distributed Problem Solving", to appear in the proceedings of IJCAI, August 1977.

APPENDIX RADIUS OF A LATTICE.

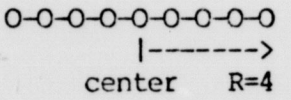
We will here give an informal derivation of the maximum distance between any two nodes in a regular lattice.

A.1. Variables and notation.

d	-order of lattice
R	-distance from the center node
N(d,R)	-Number of nodes at a distance R from the center node, in a lattice of order d
NTOT(d,R)	-Number of nodes within a distance R from the center node, in a lattice of order d
$\sum_{i=n}^{N}$	-denotes a summation with i going from n to N
$\prod_{i}^{I}$	-denotes a multiple of i sums: $\sum_{si=1}^{I} \sum_{s3=1}^{s4} \sum_{s2=1}^{s3} \sum_{s1=1}^{s2}$
droot(N)	-denotes the d'th root of N

A.2. Examples of lower orders.

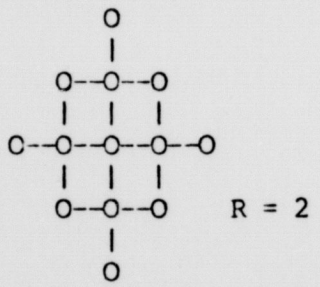
A.2.1 d=1



$N(1,R)=2$

$$NTOT(1,R)=1+\sum_{s=1}^{R} 2 = 2R+1$$

A.2.2 d=2



$$N(2,R)=N(1,R)+2*\sum_{s=1}^{R-1} EN(1,s) +2=4R$$

$$NTOT(2,R)=1+\sum_{s=1}^{R} 4s = 2(R**2) + 2R + 1$$

A.2.3 d=3

Geometry: Two mirrored pyramides, each with heigth R, and the common ground plane equal to the plane for d=2.

$$N(3,R)=N(2,R)+2*\sum_{s=1}^{R-1} EN(2,s) = 4(R**2) + 2$$

$$NTOT(3,R)=1+\sum_{s=1}^{R-1} EN(3,s) = (4/3)*(R**3) + 2(R**2) + (8/3)*R + 1$$

A.3. Iterative formulas based on geometric considerations.

Consider the coordinates describing a lattice of order d:

(X1,X2,X3,....Xd-1,Y)

When the distance from the origin is R, the absolute value sum of X1 - Xd-1 and Y must be R. The coordinates (X1,X2,..Xd-1) can describe a lattice of order (d-1), and have an absolute value sum R-|Y| (|Y| means absolute value of Y). From this we have:

1) Y=0 gives a lattice of order (d-1) where the absolute sum of X1 to Xd-1 is R.

2) Y not equal to 0 gives two possibilities for each integer between 1 and R, one for Y positive and one negative.

3) For |Y|=R the sum of X's is 0, and Y=R or Y=-R.

For the geometries we have:

$$Geometry(d,R) = Geometry(d-1,R) + 2*\sum_{s=1}^{R-1} EGeometry(d-1,s) + 2*Geometry(d-1,0)$$

N(d,R) is measure of the surface of the geometries, and therefore:

$$N(d,R) = N(d-1,R) + 2*\sum_{s=1}^{R-1} EN(d-1,s) + 2 \qquad \text{(A.3.1)}$$

and

$$(A.3.2)\quad NTOT(d,R) = NTOT(d-1,R) + 2*\sum_{s=1}^{R-1} ENTOT(d-1,s) + 2$$

or

$$(A.3.3)\quad NTOT(d,R) = 1 + \sum_{s=1}^{R} EN(d,s)$$

A.4. Generalization.

Based on (A.3.1) to (A.3.3) one can deduce:

$$(A.4.1)\quad N(d,R) = \sum_{i=0}^{d-1} E(2^{**}i) \prod_{j=1}^{d-i} E((-1)^{**}(j+1)) \; {}_{i}^{j}E1 \; {}_{i}^{R}E2$$

and

$$(A.4.2)\quad NTOT(d,R) = 1 + \sum_{i=0}^{d-1} E(2^{**}i) \prod_{j=1}^{d-i} E((-1)^{**}(j+1)) \; {}_{i}^{j}E1 \; {}_{i+1}^{R}E2$$

To simplify the expressions, we do the following approximations:

$$(A.4.3)\quad {}^{n}_{i}E1 \sim (1/i!)(n^{**}i) \quad , n>1$$
$$\sim 1 \quad , n=1$$

$$(A.4.4)\quad N(d,R) \sim 2^{**}(d-1) * \prod_{j=1}^{1} E((-1)^{**}(j+1)) \; {}_{d-1}^{j}E1 \; {}_{d-1}^{R}E2 \sim$$

$$\sim [(2^{**}d)/(d-1)!] * (R^{**}(d-1))$$

$$(A.4.5)\quad NTOT(d,R) \sim [(2^{**}d)/d!] * (R^{**}d))$$

$$(A.4.6)\quad R \sim (1/2)*droot(d!) \; * \; droot(NTOT)$$

For small values of d, we have:

d:	2	3	4	6
droot(d!)/2:	.7	.9	1	1.5

Thus

$$(A.4.7)\quad R \sim droot(NTOT)$$

gives the order of magnitude.