LBL-18132
UC-32

Lawrence Berkeley Laboratory

UNIVERSITY OF CALIFORNIA

RECEIVED
LAWRENCE
BERKELEY LABORATORY

FEB 11 1985

LIBRARY AND
DOCUMENTS SECTION

Computing Division

BACKBONE NETWORK SIMULATIONS
A COMPARATIVE STUDY

A. Engdal

June 1984

For Reference
Not to be taken from this room

LBL-18132 c.1

Prepared for the U.S. Department of Energy under Contract DE-AC03-76SF00098

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

LBL-18132

BACKBONE NETWORK SIMULATIONS
A COMPARATIVE STUDY

Alf Engdal

Computing Division
Computer Science Research Department
Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

RUNIT
University of Trondheim
N-7034 Trondheim–NTH
Norway

June 29, 1984

This work was supported by the Director,
Office of Energy Research,
Applied Mathematical Sciences program
of the U.S. Department of Energy
under Contract No. DE-AC03-76SF00098
and by the University of Trondheim, Norway

CONTENTS

1. INTRODUCTION 1

2. PAWS — A SHORT DESCRIPTION 2

3. OVERVIEW OF THE NETWORK MODELS 5

4. SHORT DESCRIPTIONS OF THE NETWORK MODELS 7

4.1. Generic Network Model 7
4.2. The PAWS Ethernet Models 7
4.3. The PAWS Token Ring Models 11
4.4. IBX10 — The PAWS IBX LANmark Model 13

5. VALIDATIONS 15

5.1. Validation — Ethernet 15
5.2. Validation — Token Ring 16
5.3. Validation — IBX LANmark 17

6. BACKBONE SIMULATIONS 19

6.1. LBL Backbone Network Alternatives 19
6.2. Ethernet Backbone Simulation 20
6.3. Token Ring Backbone Simulation 25
6.4. IBX LANmark Backbone Simulation 25
6.5. Comparative Backbone Analysis 26

7. NETWORKS WITH FINITE CAPACITY RECEIVERS 29

8. SUMMARY 31

ACKNOWLEDGMENTS 32

REFERENCES 33

BACKBONE NETWORK SIMULATIONS
A COMPARATIVE STUDY

Alf Engdal

Computing Division
Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

1. INTRODUCTION

Local Area Networks (LAN's) interconnected by some sort of high-capacity "backbone" network are emerging as a common overall network concept for large institutions. There are many alternative technologies available to establish backbones: baseband networks such as Ethernet, broadband, based broadcast networks, token ring networks, and an increasing number of digital PBX's that have LAN interfaces to high bandwith switched data channels. The LAN's will connect to the backbones via gateways. The gateways are not readily available off the shelf, partly because of the lack of sufficient standards. There is however a extensive work going on to establish such gateways for different LAN and backbone technologies.

The present work focuses on the backbone part, although many of the models may be equally applicable for LAN simulation. The reason why we have focused on the backbone is the present LBL effort to plan a laboratory-wide, resource-sharing network that shall provide communication links among all Laboratory facilities. A backbone facility for LAN's will be an essential part of such a network. The results presented herein will hopefully be an aid in choosing between different backbone alternatives.

PAWS—Performance Analysts Workbench System—has been used to do the simulation work reported here. PAWS is a general tool for simulating computer and communication systems. It was developed some years ago, but has been greatly improved recently. One aspect of this work has been to gain experience with the use of the new version of PAWS for computer network simulations.

A small effort has been devoted to the simulation of networks with "finite capacity receivers." This is due to earlier experiments conducted at LBL [5], which have shown some peculiar characteristics of local networks, even for small loads.

This work was done as part of the authors sabbatical from RUNIT, The Computing Centre at the University of Trondheim, Norway.

2. PAWS — A SHORT DESCRIPTION

PAWS is based upon a pictorial construct for modelling information processing systems, called an INFORMATION PROCESSING GRAPH, hereafter abbreviated IPG. An IPG consists of *nodes* connected by *edges*. Each node in the graph represents a resource (e.g., processor, memory, network link, bus), a routing decision, or a computational step. Each edge in the graph represents some form of information flow from one node to another along the direction of the edge. Nodes are given unique names. Edges have *labels* denoting the form of information flow along the edge.

The basic entities of information that get processed and routed are *transactions*. A transaction in general represents the data on which the nodes operate. For computer network simulations, a transaction will most often be a message or a packet. Several transactions may be simultaneously active (being operated upon) at different nodes of the IPG.

There may be different types of transactions — CATEGORIES — and different PHASES within each category. This is used to handle and route transactions differently through the IPG.

There may be several edges leading from one node. In that case, each edge is labelled with the probability that a transaction of a specific category and phase will choose that specific edge.

A PAWS program can manipulate three types of variables: integer and real scalar variables, global variable tables, and local variable tables. Local here means local to a specific transaction. The scalar variables are global, that is, they are accessible in each node for the handling of any transaction category.

A PAWS program has the following structure:

```
<program> :: =
    {OPTIONS EFORMAT ; }
    DECLARE
        {INTEGER      <int-scalar>   ... ; }
        {REAL         <real-scalar>  ... ; }
         NODES        <node>         ... ;
         CATEGORIES   <category>     ... ;
        {TOKENS       <token>        ... ; }
        {MEMORIES     <memory>       ... ; }
    TOPOLOGY
        <from-node>  <to-node> <cat-phase> <prob>;
        :
    DEFINE
        <node-def> ;
        :
    {INITIAL    <pop> ; }
    {STATISTICS <report> ; }
    RUN
         <run-stmt>
        :
        ;
    END;
```

PAWS supports the following probability distributions for transaction generations, service times etc.:

-----DISTRIBUTIONS

EXPO(m)
CONSTANT(c)
HYPER (m,s)
ERLANG (m,s)
UNIFORM (a,b)
EMPIRICAL (p1,a1,b1,p2,a2,b2.....)

and the following queueing disciplines:

-----QUEUEING DISCIPLINES

FCFS (First Come First Serve)
RRFQ (round Robin Fixed Quantum)
PS (Processor Sharing)
PRIORITY (non-preemptive)
PREEMPTIVE PRIORITY
LCFSPR (Last Come First Serve Preemptive Resume)
POLLING
DELAY (only at service nodes)

The following node types are being used in the present network models:

SERVICE	Used by a transaction to acquire a server.
ALLOCATE	Used by a transaction to acquire TOKENS.
RELEASE	Used by a transaction to release tokens.
SET	Used by a transaction to alter the POWER of one or more of the node types listed above. POWER may be the processing capacity of a CPU node or the number of tokens held by a network interface.
SPLIT	Used by a transaction to create new transactions.
FORK	Similar to SPLIT, but the transaction requesting a 'fork' does not leave the FORK node until all the created transactions have reached the same common JOIN node at which time the 'parent' transaction replaces its 'children' and proceeds from the JOIN node.

JOIN	(see above)
SOURCE	Used to periodically generate new transactions.
SINK	When a transaction reaches a SINK node it departs form the system forever.
BRANCH	Reroutes arriving transactions to other nodes.
COMPUTE	Used by a transaction to interrogate and alter variables. There is a set of different operations available for assignment operations, sequence control and simulation control.

For more details on PAWS, read [8].

3. OVERVIEW OF THE NETWORK MODELS

PAWS models exist for the following networks:

- Ethernet (models ETHER10/13, ETHER20, ETHER30/31)
- Proteon proNET® (models RING10, RING30)
- InteCom® IBX® LANmark®/InteCom (model IBX10)

The **Ethernet** is a baseband coaxial-cable based LAN that operates at 10 Mbps over a distance of up to approximately 2,500 meters. It is of the general broadcast bus class of LAN's and utilizes an access arbitration technique now commonly called CSMA/CD (Carrier Sense Multiple Access with Collision Detection). See [6] for the specific details of Ethernet.

The **Proteon proNET** is a baseband LAN that operates at 10 Mbps over a distance of a few kilometers. It is of the general ring class of LANs and utilizes an access arbitration technique commonly called token passing. See [9] for the specific details of proNET.

The **InteCom IBX LANmark** is a digital PBX based LAN that operates at 1 Mbps over a distance of up to approximately 15,000 meters. It provides gateway connections to distributed Ethernet Segments allowing a packet generated on one Ethernet to appear on the addressed distination Ethernet. It is of the general star configuration class of LAN's and utilizes an access arbitration technique of finite buffer queueing with no guaranteed delivery. See [7] for more information on LANmark.

Where necessary, it will be stated which version of the network is being modelled, or which features of the network are being included in the models.

In addition to modelling different networks, the models differ on the following features:

heterogeneous/homogeneous transmitters
include/do not include finite capacity receivers

With *homogeneous* transmitters the total load on the network is considered as coming from one source and having a specific arrival distribution, which may be any one of the distributions available under PAWS: Exponential, Hyperexponential, Erlang, Uniform, Constant or Empirical.

With *heterogeneous* transmitters there may be different types of transmitters or transmitter groups, each with its own arrival distribution, chosen from those mentioned above.

All models may simulate different packet length distributions.

Finite capacity receivers means that the model comprises one or more receivers that are single-buffered devices unable to receive another packet while they are busy transferring a received packet from the receiver interface into main memory. Messages or packets arriving at a busy receiver will normally be counted, so as to keep track of the percentage of lost messages.

The existing models conform to these features as follows:

	homogeneous xmitters	heterogeneous xmittersi	receiver delay
ETHER10/13		X	
ETHER20	X		
ETHER30/31		X	X
RING10		X	
RING30		X	X
IBX10		X	

Ethernet is modelled according to the IEEE Standard for CSMA/CD [6]. There is one exception. ETHER13 models the prototype 3 Mbps Ethernet corresponding to the one used to measure performance at Xerox PARC [3]. ETHER13 is being used for validation purposes.

The Ethernet models have been run with different collision windows (CW). CW is the elapsed time from the start of a packet transmission until the bus is acquired, i.e., the time "slot" in which a collision may occur. CW is not part of any standard, but is entirely dependent upon the physical size of the network, including the number of repeaters used. The maximum value of CW, according to IEEE 802, is 49.89 μsec for a 2.5 km network with 4 repeaters. For a single segment net, CW = 3.16 + 8.32L, where L is the cable length measured in kilometers. It should be noted that CW is not the same as the SlotTime (ST), which is part of the standard. See Section 5.1 for more about ST.

The transfer delays between memory and network interfaces that are used in the ETHER30/31 and RING30 models are taken from measurements performed at LBL [5].

The Token Ring is modelled according to specifications for the 10 Megabit/sec Proteon Token Ring Network being used at LBL, but with some modifications as described in Section 6.1 [9].

The IBX model features on-the-fly-routing, packet fragmentation and packet broadcast. On-the-fly means that if the specified output channel is available at the time the Router starts to receive a packet, the output will start immediately, reducing the total delay imposed by the IBX for that specific packet by nearly 50%. This is an inherent feature of IBX LANmark. Fragmentation is necessary within IBX because the maximum IBX frame length is 266 bytes as compared to the 1512 byte packets that are accepted from the distributed Ethernets interconnected by the IBX.

An IBX input port may send broadcast packets for two different reasons: a) the packet has a broadcast address, or b) the packet has an address that is unknown to the input port at present. The broadcast is actually performed by the central Router. Both fragmentation and broadcast may be set in the model to happen to a specified fraction of the packets, independent of each other.

4. SHORT DESCRIPTIONS OF THE NETWORK MODELS

The network models will be described by their Information Processing Graphs (IPG's). IPG's are drawn using triangles, squares, diamonds and circles to represent different types of IPG nodes. *Triangles* are used to depict routing nodes, i.e., SOURCE, SINK, SPLIT, FORK, JOIN and BRANCH. *Squares* depict arithmetic nodes, like COMPUTE. Two different symbols are used for resource management nodes: *diamonds* for RELEASE (tokens) and SET (power); *circles* for SERVICE and ALLOCATE. Note that circles in the IPG's will always have a queue associated with them. The queue may, however, have a maximum size of 1 depending on the model. An example is the Ethernet BUS service node to which access is controlled by the collision detect mechanism so as to prevent more than one message from entering the BUS node at any one time.

4.1. Generic Network Model

The generic network model is depicted in Fig. 4.1.

The LOAD part differs somewhat for the 3 different model categories, and is to some extent dependent upon the network being modelled.

The NETWORK part is totally different for each different network type (i.e., Ethernet, Ring and IBX models.)

The NETWORK EXIT part is usually very simple, either one exit node, or a number of EXIT nodes corresponding to the number of LOAD nodes.

The RECEIVER part is only present in the ETHER30, ETHER31 an RING30 models.

4.2. The PAWS Ethernet Models

The IPG's for ETHER10 and ETHER30 are given in Figs. 4.2 and 4.3, depicted for a limited number of transmitters/receivers. Enhancing the configurations is only a question of adding identical transmitter/receiver parts. This is quite easy in Paws, and is usually done by changing dimension fields in the programs.

A short description will be given for the different models.

ETHER10

The 5 NODE nodes simulates 5 transmitters. Packets may be routed through each one of these according to packet generation in MSOURCE and the setting of relevant parameters in SETUP. Packet arrivals may have identical or different distributions for each one of the NODE's, but in all later simulations described in this report the arrival distributions are similar for all NODE's.

The NETWORK part has basically 3 different components: *collision detection and resolve, bus* and *backoff system.* Collision detection is done by holding all packets in CDS for a time equal to the collision window. As a packet leaves CDS, CRC checks if there are still more packets in CDS. If so, a collision has occurred and all involved packets are routed through the backoff system. If not, the bus is acquired and the packet "enters" the bus node after passing through DEFER, which prevents all other packets from entering CDS until BUS has finished servicing the packet, i.e., the packet is transmitted. The backoff system comprises N BACKOFF service nodes, where N is the number of transmitting nodes (NODEs). The

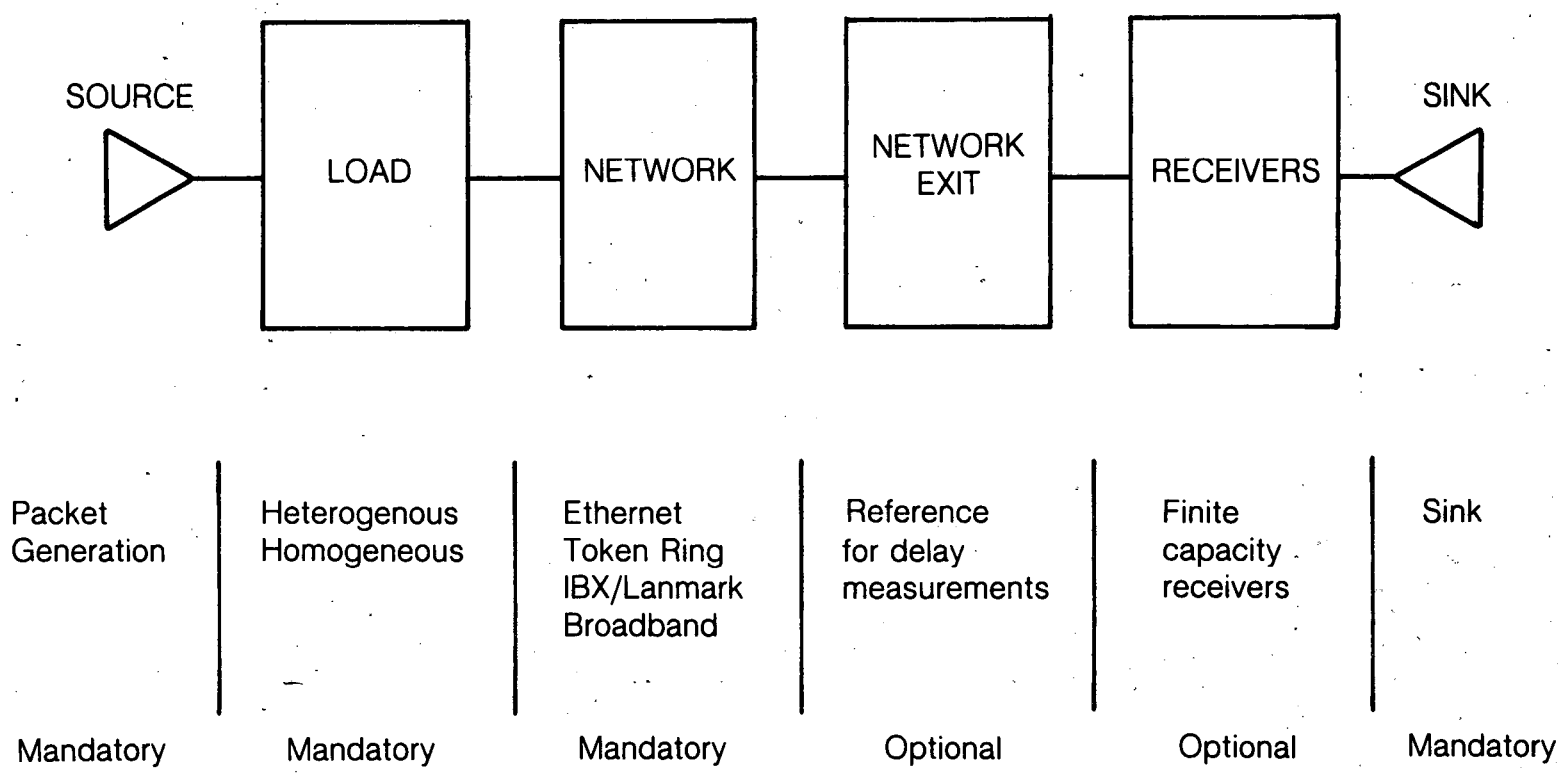
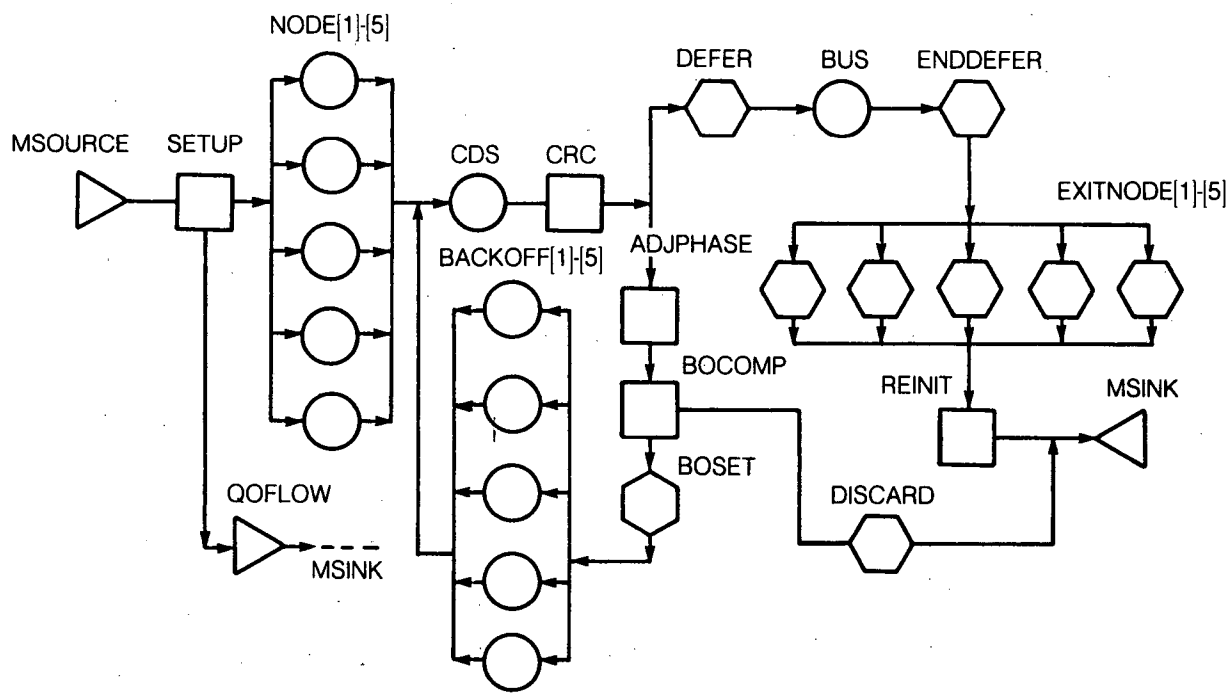
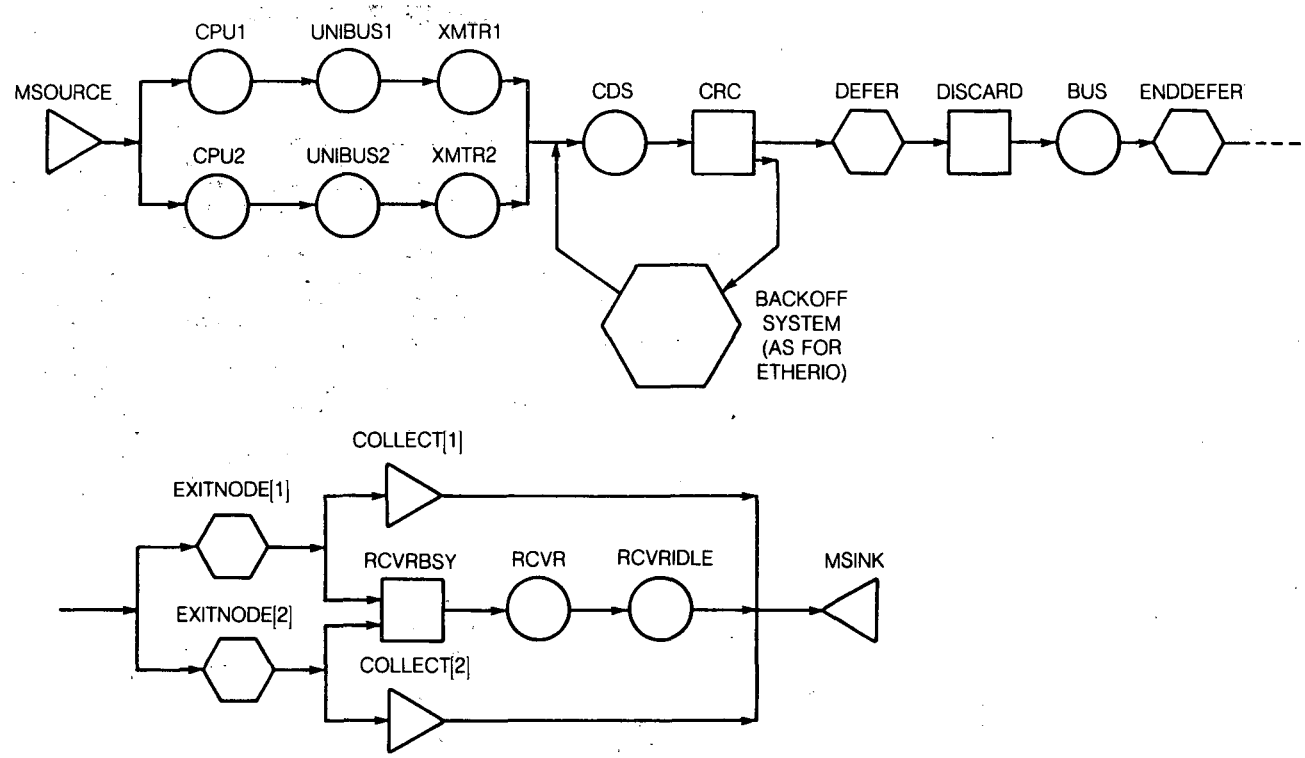


Fig. 4.1. Generic simulation mode.



NODE NAME	TYPE	FUNCTION
MSOURCE	SOURCE	Packet source.
SETUP	COMPUTE	Sets certain packet local variables
QOFLOW	BRANCH	Reference to count lost packets due to queue overflow
NODE[1]–[5]	ALLOCATE	Transmitting nodes
CDS	SERVICE	Detects collisions
CRC	COMPUTE	Resolves collisions, routing colliding packets to backoff system
ADJPHASE	COMPUTE	Readjusts phase after collision rerouting
BOCOMP	COMPUTE	Computes backoff delay. Delay depends upon number of rexmissions
BACKOFF[1]–[5]	SERVICE	Backoff nodes, hold packets being backed off
DEFER	SET	Sets bus acquired by preventing more packets from entering CDS
BUS	SERVICE	Bus
ENDDEFER	SET	Releases bus by admitting more packets to enter CDS
EXITNODE[1]–[5]	RELEASE	Releases transmitter token acquired at NODE
REINIT	COMPUTE	Resets some variables
DISCARD	RELEASE	Releases transmitter tokens held by discarded packets
MSINK	SINK	Sink

Fig. 4.2. ETHER10 — Ethernet PAWS mode.



NODE NAME	TYPE	FUNCTION
MSOURCE	SOURCE	Packet source.
SETUP	COMPUTE	Sets certain packet local variables
CPUx	ALLOCATE	Allocates transmit tokens, initially holds 1
UNIBUSx	SERVICE	Requests transfer of packets to transmit interface nodes
XMTR1	BRANCH	Actually dummy as part of model
CDS	SERVICE	Detects collisions
CRC	COMPUTE	Resolves collisions, routing colliding packets to backoff system
DEFER	SET	Sets bus acquired by preventing more packets from entering CDS
DISCARD	COMPUTE	Tests receiver availability. If receiver busy, changes phase to make packet pass by receiver
BUS	SERVICE	Bus
ENDDEFER	SET	Releases bus by admitting more packets to enter CDS
EXITNODE[1]–[5]	RELEASE	Releases transmitter token acquired at CPU
REVRBUSY	COMPUTE	Sets variables indicating receiver busy
RCVR	SERVICE	Request receiver service, i.e., transfer of packet to main memory
RCVRIDLE	COMPUTE	Resets variables indicating receiver busy
MSINK	SINK	Sink

Fig. 4.3. ETHER30 — Ethernet PAWS model with finite capacity receivers.

BACKOFF nodes delay the packets for a "backoff" time, which is computed in BOCOMP according to the Ethernet algorithms for such delay. After backoff, the packets try once more to acquire the bus, by entering CDS. Each packet may be involved in a number of collisions, but is discarded after 16 transmission attempts. Note that the time spent in CDS has to be accounted for when modelling actual transmission time, since it is actually part of the transmission of the whole packet.

ETHER20

The ETHER20 model is similar to ETHER10, but with single NODE and EXITNODE. With a large backoff system it may be used to simulate near "infinite" population.

ETHER30

The LOAD part in ETHER30 includes the CPU and CPU-to-interface bus. This bus node is called UNIBUS for some obvious reasons at LBL. Instead of the EXIT nodes giving the transmit tokens back to the NODE nodes (or the transmitters) as is done in ETHER10, tokens are now returned to the CPU's, which then have to intiate transfer of the next message via UNIBUS to the transmitter. By initially having 1 token at the CPU node, we actually simulate single buffered transmitters.

The RECEIVER part models a single buffered receiver. A single buffered receiver is unable to receive another packet while it is busy transferring to main memory a packet that is already received. Packets entering the bus while the corresponding receiver is busy, are discarded, being routed through a counting node that keeps track of the number of discarded packets.

This requires one change to the modelling of the NETWORK part itself as compared to the two preceeding models. This is due to the fact that a packet is being discarded if the receiver is busy *when it receives the start of the packet.* The test for busy receiver has therefore to be done in front of the BUS node as depicted.

4.3. The PAWS Token Ring Models

RING10

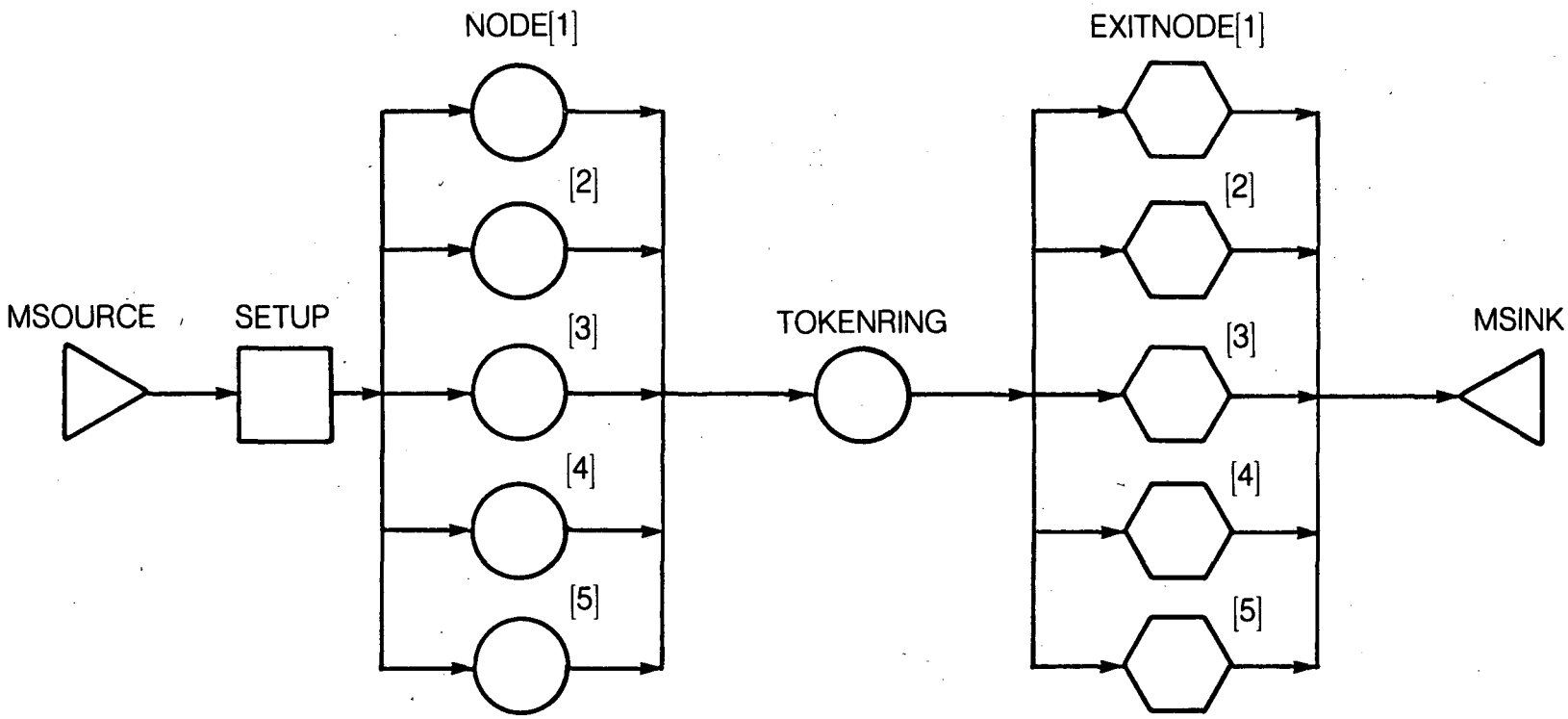
The IPG for RING10 is depicted in Fig. 4.4.

This is a very simple model. The only thing to notice is that the token passing is simulated using a POLLING queueing discipline in front of the TOKENRING node. Packets are inserted in this queue according to their node number; packets from node 1 in the first "slot" of the queue, packets from node 2 in the second, and so on. Packets are given service in sequence according to slot number, not according to time of arrival.

No delay has been assumed to pass token between nodes.

RING30

The difference between RING30 and RING10 is the same as the difference between ETHER30 and ETHER10. The transmitter and receiver part for RING30 equals that of ETHER30.



NODE NAME	TYPE	FUNCTION
MSOURCE	SOURCE	Packet source
SETUP	COMPUTE	Sets certain packet local variables
NODE[1]–[5]	ALLOCATE	Transmitting nodes
TOKENRING	SERVICE	Models the token ring, by having associated with it a POLLING queue
EXITNODE[1]–[5]	RELEASE	Releases transmitter token acquired at NODE
MSINK	SINK	Sink.

Fig. 4.4. RING10 — Token ring PAWS model IPG.

4.4. IBX10—The PAWS IBX LANmark Model

The IPG for the IBX LANmark model is given in Fig. 4.5. Only one input and one complete output channel is shown. Each node as shown in the figure is described below.

ISOURCE generates input packets of type MSG. MSG type packets may be routed with a specified probability to FRAGMENT, a node of type FORK. FRAGMENT holds back the originating MSG packet, and spawns a number of fragments, F. The number of fragments is specified as one of the available PAWS distributions.

The preceeding mechanisms model the input part of an IBX DOB (Data Option Board).

ACQICHAN controls access to the input channel, holding one token initially. Packets may be routed on the fly, in which case they skip the model's input channel BBI, and advances directly to the output channel BBO. One must take care in reading simulation outputs on this point. What actually happens in an IBX on-the-fly routing is that the packet will exist simultaneously on the input and output channel. The utilization figures for BBI in a simulation output will therefore be too low and should actually equal the utilization figures for BBO.

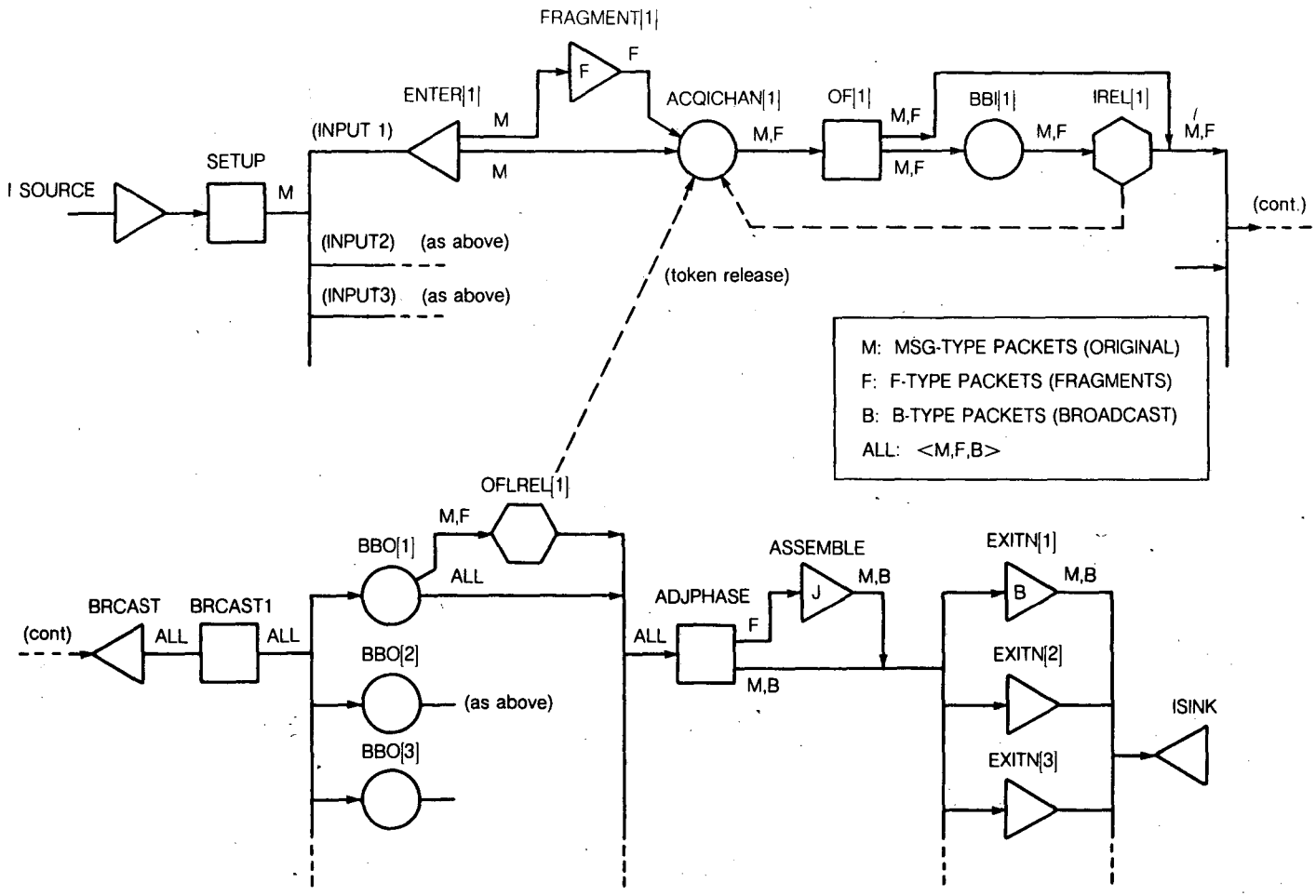
Both MSG and F-type packets may be broadcast (BM or BF). This is modelled in nodes BRCAST and BRCAST1. BRCAST generates N-1 packets of types BM if the original packet is of type MSG, and type BF if original packet is of type F. BRCAST1 selects output channels for B-type packets in ascending order, modulo N. This should be a good model of the IBX broadcast function. The only weak part in the model, is the combination of on-the-fly and broadcast, because it is not clear how this is done within IBX. In the model, no on-the-fly will happen for any of the broadcast packets if originating message was not routed on-the-fly. This may be an incorrect assumption for the IBX.

The input channel access token is released from IREL for ordinary routing or from OFLREL for on-the-fly routed packets.

Fragments are assembled back into original packets in the ASSEMBLE node of type JOIN (J) before the original packets advance to EXITN, which is the reference point for delay measurements.

ACQICHAN models the input channel grant mechanism of the IBX IM Buffer Board, BBI is the input channel, and BBO the output channel. OFL models the on-the-fly routing mechanism within the IBX LANmark Router.

ASSEMBLE models the output part of a DOB, which assembles fragments to whole Ethernet packets.



NODE NAME	TYPE	FUNCTION
ISOURCE	SOURCE	Packet source
SETUP	COMPUTE	Sets certain packet local variables
ENTER	BRANCH	Branches off packets to be fragmented
FRAGMENT	FORK	Fragments "long" packets, by holding original and sending off a specified number of fragments
ACQICHAN	ALLOCATE	Allocates input channel tokens
OFL	COMPUTE	Performs on-the-fly routing if output channel available
BBI	SERVICE	Requests transmission on input channel
IREL	RELEASE	Releases input channel tokens
BRCAST	SPLIT	Generates broadcast packets. Broadcast packets are routed uniformly to each one of the output channel nodes (BBO)
BRCAST1	COMPUTE	Selects output channels for broadcast packets
BBO	SERVICE	Requests transmission on output channel
OFLREL	RELEASE	Releases input channel tokens for on-the-fly packets
ADJPHASE	COMPUTE	Readjusts phase changed to invoke on-the-fly rerouting
ASSEMBLE	JOIN	Assembles fragments to whole packets
EXITN[1]-[5]	BRANCH	Reference points for delay measurements
ISINK	SINK	Sink

Fig. 4.5. IBX10 — IBX LANmark PAWS model IPG.

5. VALIDATIONS

All models have been validated to the extent that they do not produce results that are obviously contradictory to what they model, or to any part thereof. The flow of transactions through the different IPG nodes is consistent, no nodes are being utilized above their capacity and so on. As an example of such consistency control, the Ethernet models are constructed such that there should be no more than 1 message at the BUS IPG node at any one time. All the present Ethernet models act according to this, while an attempt to create a model with more complicated collision handling resulted in a model that occasionally produced 2 messages on the bus, indicating there was something wrong with that model.

For further validations there are two different methods available:

1. Setting up simulation environments that can be solved analytically
2. Simulating environments that have been subject to real experiments, and where measurements and results are available.

The Ethernet and Ring models have been subject to both types of validations. No measurements exists as far as the author knows for the IBX LANmark. However, a validation attempt has been made, running a simulation with simple arrival rates and packet length distributions.

5.1. Validation—Ethernet

There has been numerous efforts to construct analytic models for Ethernet local networks. One of the most renowned articles on the matter is the one by Tobagi and Hunt [2]. The analytic models usually have some constraints that make a simulation of the same models somewhat difficult. [2] describes a model of a network with infinite population, impossible to simulate by PAWS or any other simulation tool. Exact reproduction of the analytic results will therefore be impossible, but one should expect a qualitative resemblance between the simulation model and such results.

The measurement reference used are the results presented by Shoch and Hupp in [3], especially the part where they measured performance characteristics under high load conditions. This was done by creating artificially high load from a limited number of hosts on the Xerox Parc 1800 foot long with 120 Ethernet hosts attached to it. One piece of information is however lacking from the article, the SlotTime for that network. The SlotTime is a very crucial Ethernet variable for many reasons:

- it is an upper bound on the acquisition time of the medium
- it is an upper bound on the length of a frame fragment generated by a collision
- it is the scheduling quantum for retransmission.

The slottime in the standardized IEEE Ethernet is 51.2 μsec [6]. In the first Ethernets as documented by [1], it was 16 μsec. It is believed that the Xerox Parc network used 16 μsec, and this is therefore used in the validation runs. It was also a 3 Mbps network as compared with 10 Mbps for the standard.

The experiments described in [3] have the following variables:

- number of hosts involved
- offered load, that is the total traffic if the network had infinite capacity
- packet length (hereafter denoted PL and given in bytes)

What is measured is bus utilization and fairness. Bus utilization is the percentage of the time the network carries useful traffic. Fairness is how much of this time is being used by each one of the hosts.

For offered load below 80-85% the utilization will roughly equal the load. This is also the case for the simulation model (ETHER13), with utilization above 49% for 50% load and all PL above and including 64 bytes, and above 79% utilization for an 80% load.

For 100% load, XEROX measured a utilization ranging from 83 to 94% for PL ranging from 64 to 512 bytes. The corresponding values from an ETHER13 simulation is 91–94%, with 5 hosts. Reference [3] does not specify the number of hosts in their measurements, but there is reason to believe it is higher than 5 (probably 10 or 20). This could account for some of the difference, since [3] also shows that utilization will decrease as number of hosts increases. The interface delays will also play a part, as mentioned earlier.

The simulated values are closer to the analytic values computed, which according to [3] lie in the region above 95% for 5 hosts and for all packet length values above 64 bytes.

Both measurements and simulation show a high degree of fairness on an Ethernet. For a 94% utilization, Xerox measured individual throughputs ranging from 9.3–9.6% for 10 hosts. The corresponding simulated values for 5 hosts is 18.2–19.3%, a slightly larger gap, which would probably be reduced if the simulation was run for a larger amount of time.

Going from the Xerox 3 Mbps Ethernet with SlotTime (ST) of 16 μsec, to an IEEE 10 Mbps with ST=51 μsec, one would according to [2] expect a significant reduction in utilization given as percentage of bus capacity, at least for small packets. This is also the case. ETHER10 yields the following utilization values for an IEEE Ethernet with 5 hosts:

	Packet Length in bytes	
	PL=64	PL=512
LOAD=50%	49	49
LOAD=80%	64	75
LOAD=100%	64	84

These results show that for PL=64, the maximum utilization according to simulation is 64%. For loads above this value, the network will be continuously queued, and utilization will only vary with number of hosts for constant packet sizes.

What these validation runs show is mainly that the simulation models for Ethernet at least behave qualitatively as one would expect. The quantitative measures are within acceptable limits so as to believe that simulation results will give a useful picture of the behaviour of a real Ethernet.

5.2. Validation—Token Ring

The token ring model is quite simple, as is the concept of the token ring itself. Using the POLLING queueing discipline of PAWS one almost gets the token ring model for free, and

there is not very much need for a large validation procedure. It has been shown, however, that the model produces a linear increase in bus utilization for loads increasing from 0 to 100%, thereafter staying at 100% for higher loads. This is the expected behaviour for such a network. The bus capacity is fairly shared between the hosts.

The simulation of the LBL Receiver Delay experiment described in Section 7 is also taken as part of the validation of the Token Ring model.

5.3. Validation—IBX LANmark

The validation of IBX10 has been quite difficult, and cannot at present be said to be adequate. First, the author has not had access to any measurement results. Secondly, quite simple models and load distributions will give environments for IBX that are not easily solvable by analytic means.

What is done then is to run a simulation with simple distributions, starting out with no broadcast and no fragmentation. Thereafter a probability is introduced for broadcast, and eventually for both broadcast and fragmentation. The results are shown in Table 5.1

A few comments to the table:

- The packet counts are as expected. For pB = pF = 0.2, 994 type MSG packets generate 1019 fragments, which is close to 994 * 0.2 * 5. The remaining 785 MSG packets and the fragments yields 1416 broadcast packets, close to (785 + 1019) * 0.2 * 4 = 1443. Note that this only shows that the model is working according to its own premises, and gives a correct picture of IBX, as long as the presuppositions for the model are correct.
- The delays can be shown to have reasonable values. Minimum delay is 1 msec for 1000 bits packets routed on-the-fly as expected. Note however that the model assumes 0 propagation delays and no delays due to internal processing within IBX. These delays should be included in an improved model.
- An experiment corresponding to this simulation should have been set up for a real IBX to get a proper validation of the model. It should be quite feasible to do such an experiment, if one has access to an IBX with LANmark installed.

IBX10 turns out to be the most complicated of all the models. Even with only 5 I/O channels it requires significant amounts of run-time. For IBX you increase bandwith as you increase the configuration, as opposed to Ethernet and Token Rings. The hypothesis was therefore that adding I/O channels to the simulation model would give little new information. Simulations with 2, 5 and 10 I/O channels confirmed this. There were some differences between 2 and 5, but only minor differences between 5 and 10. Simulating 20 I/O channels, which is of special interest in the backbone simulations, is therefore done by simulating 5 channels, and then making up for the differences where necessary by actually multiplying queue lengths etc., by 4.

Table 5.1 IBX VALIDATION

No. of I/O channels: 5
Packet size: 1000/5000 bits
Fragment size: 1000 bits
No of fragments pr. packet: 5
Arrival rate: UNIFORM(3,7)
==> 200 packets/sec/channel on average
Simulation time: 1 sec.

Broadcast probability Fragmentation probability (% of packets of size 5000 bits)	pB = 0.0 pF = 0.0	pB = 0.2 pF = 0.0	pB = 0.2 pF = 0.2
PACKET COUNTS			
Input, type MSG	998	990	994
Transmitted, type MSG	997	987	785
Transmitted, type B	0	753	1416
Transmitted, type F	0	0	1019
Transmitted, ALL	997	1740	3220
PERCENTAGES			
% broadcast packets	0	43	44
% fragment packets	0	0	32
ON-THE-FLY STATISTICS			
No of on-the-fly (MSG)	856	728	301
-------"------- (F)	0	0	310
-------"------ (ALL)	856	728	611
% on-the-fly (MSG)	86	74	38
% on-the-fly (F)	0	0	30
% on-the-fly (ALL)	86	74	34
QUEUE LENGTHS			
Max inp. queue length (MSG+F)	1	4	28
Max outp. queue length (ALL)	2	4	15
UTILIZATION (%)			
Mean BBO utilization	20	35	64
DELAYS			
Mean delay (MSG)	1.16	1.88	4.41
Min. delay	1.0	1.0	1.0
Max. delay	3.0	4.3	18.9

6. BACKBONE SIMULATIONS

6.1. LBL Backbone Network Alternatives

The basic purpose of the simulations done so far has been to get an impression of capacity and performance for 3 different hypothetical LBL backbone networks: Ethernet, Proteon proNET 10 Mbps Token Ring, and InteCom IBX LANmark. These backbone networks are hoped to interconnect up to 20 Ethernet segments, a realistic environment for LBL.

Two other very interesting backbone alternatives have not been simulated under this study: the broadband local area network and the new 80 Mbps token ring network being developed by Proteon, Inc. These should be included if the backbone network simulations are to be continued. All networks are packet switched. The packet sizes are taken to have a distribution conforming to the distribution measured on the present LBL token ring local network. The arrival rates simulated are exponentially distributed with means of 4, 10 and 20 msec respectively, corresponding to Poisson packet sources with mean throughput of 250, 100 and 50 packets per second.

The packet length measurements at LBL showed the following distribution:

Table 6.1

Bin #	Range	Fraction of packets
1	1- 70	0.4075
2	71- 100	0.0094
3	101- 130	0.4391
4	131- 300	0.0108
5	301- 460	0.0253
6	461- 550	0.0113
7	551- 600	0.0149
8	601- 660	0.0413
9	661-1100	0.0404

To get frame lengths, which is what we are interested in, one has to add for

Ethernet: 27 bytes
Token Ring: 17 bytes
IBX: 19 bytes.

For Ethernet one has to also remember the minimum frame length, which is 512 bits, and for IBX the maximum packet size, which is 266 bytes. The IBX packet size distribution applied is further described in Section 6.4.

NOTE 1: By a misunderstanding the measured values have been interpreted as packet sizes while they are actually frame sizes on the present LBL Token Ring network. The frame overhead is 6 bytes for that network. With an average packet size of 160 bytes, the error is below 4%.

NOTE 2: The frame overhead used for the following Token Ring simulations is 17 bytes. This is more according to expected future standards than the present frame

overhead of 6 bytes. Future standards will probably specify 12 bytes solely for addressing [10].

The following sections describe independent simulations of the 3 network types, followed by a comparative study of the behaviour of the networks under equal input load.

The delays listed in the tables include both queuing times and transmission times. For Ethernet and Token Ring there is one queue, namely the input queue. The IBX model has two internal queues as is described in Section 6.4. The IBX and Token Ring models are taken to have infinite input queue capacities, while there is a maximum input queue capacity for Ethernet. If the input queue is overflowed, packets are discarded. The queue limits simulated for Ethernet is 50. For comparison, note that the maximum input queue lengths for the Ring and IBX simulations stay below 20 for all simulated loads.

6.2. Ethernet Backbone Simulation

Results are shown in table 6.2a for load of 50, 100, (150) and 250 packets/sec, and for a 10 second simulation time.

The main thing to notice is that delays and queue lengths increase moderately from 50 to 100 packets/sec (p/s), but that there is a large increase for 150 p/s. For 250 p/s, the packet loss is 11.5%, mainly due to input queue overflow. This has been measured to 2% for 200 p/s, while it is 0% for 150 p/s. However, for all loads above 100 p/s, there is a small loss due to maximum number of backoffs. According to these simulations there is reason to believe that 150 p/s is close to the capacity for an Ethernet backbone, if the packet sizes conforms to the distribution measured on the present LBL local network. 100 packets/sec should, however, be well within Ethernet capacity.

The collision statistics are worth a closer look, and require some additional comments. Note first that the number of packets involved in a collision may include a single packet colliding more than once, while the last line in the collision statistics says something about the history of a single packet that has eventually been transmitted to the receiving side. That is why 6% of the number of transmitted packets (606) is not equal to the number of colliding packets (1054). In fact we see that many of the packets collide more than once. The reason is probably that if two packets collide on Ethernet, and it is the first collision for both, then there is 50% chance for a new collision on the first retransmission attempt. This is due to the algorithm for computing backoff times.

It is also interesting to look at the backoff distribution given in Table 6.2b below. The numbers denote the number of packets that have been through 0,1,2,....16 backoffs respectively. Sixteen is the maximum number, after which a packet is being discarded. Even though there is a lot of collisions for the highest load, 60% of the packets are still transmitted on the first try. The corresponding numbers for 50 and 100 packets/sec load are 96 and 85%.

Table 6.2a ETHERNET BACKBONE SIMULATION

Number of gateways: 20 (with equal load)
Transmission speed: 10 Megabit/sec
SlotTime: 51 microsec
Collision window: 49 μsec (maximum = 49.89 μsec)
Packet size: EMPIRICAL (distribution according to LBL measurements)
Average packet size: 178 bytes (1420 bits)
Arrival rate: EXPONENTIAL(Mean = 1/(number of packets per sec))
Max. input queue length: 51 packets
Simulation time: 10 sec.

LOAD (packets/sec)	50	100	(150)	250
PACKET COUNTS				
No. of packets transmitted	9893	19873	(30118)	44310
Count per gateway (avg.)	495	994	(1506)	2216
UTILIZATION (%)				
Bus utilization	15	30	(45)	66
Bandwith share—min.	.67	1.41	(2.19)	3.05
Bandwith share—max.	.80	1.61	(2.38)	3.59
DELAYS (msec.)				
Mean delay	.21	.53	(13.1)	81
Max. delay	3.5	107	(266)	526
Standard deviation	.30	2.3	(30)	84
% with delay < 2 msec	99.6	94.5	(67)	16
QUEUES				
Mean input queue length	.001	.01	(35)	20
Max input queue length	2	15	(50)	51
COLLISION STATISTICS				
No. of collisions	1319	2773	(20035)	21992
No. of packets involved (incl. retransm. attempts)	2759	6360	(48608)	50792
Packets/collision (mean)	2.09	2.29	(2.40)	2.31
% of packets colliding	12	33	(54)	40
LOST PACKETS (%)				
Due to inp. queue over flow	0	0	(0)	11
Due to max. no. of backoffs	0	0	(.02)	.5
Total	0	0	(.02)	11.5

Table 6.2b ETHERNET BACKBONE SIMULATION. BACKOFF DISTRIBUTION.

Simulation parameters, see table 6.1

The numbers given are the number of packets that have had n backoffs in the Ethernet backbone simulations.

LOAD (packets/sec) No of backoffs	50	100	250
0	8724	13231	26369
1	336	2015	9450
2	371	1636	3327
3	257	1236	1724
4	135	735	843
5	53	486	507
6	14	292	362
7	3	151	318
8		58	286
9		16	243
10		13	237
11		2	150
12		1	157
13			131
14			74
15			68
16			64
>16			227

Table 6.3 TOKEN RING BACKBONE SIMULATION

Number of gateways: 20 (with equal load)
Transmission speed: 10 Megabit/sec
Packet size: EMPIRICAL (distribution according to LBL measurements)
Average packet size: 174 bytes (1420 bits)
Arrival rate: EXPONENTIAL(Mean = 1/(number of packets per sec))
Simulation time: 10 sec.

LOAD (packets/sec)	50	100	250
PACKET COUNTS			
Packets transmitted	10061	20055	50320
Average count per gateway	503	1003	2516
UTILIZATION (%)			
Bus utilization	14	28	70
Bandwith share—min.	0.64	1.31	3.38
Bandwith share—max.	0.77	1.49	3.62
DELAYS (msec.)			
Mean delay	0.17	0.21	0.55
Max. delay	1.7	2.2	12.5
Standard deviation	0.20	0.25	0.74
% with delay < 2 msec	100	100	95.7
QUEUES			
Mean input queue length	.001	.001	.02
Max input queue length	2	2	5
Max packets waiting for token	5	8	17

Table 6.4 IBX LANmark BACKBONE SIMULATION

No. of gateways (=no. of IBX 1Mbit I/O channels): 5
Packet size: EMPIRICAL (acc. to LBL measurements)
Broadcast: NONE
Fragment size: EMPIRICAL (acc. to LBL measurements)
No. of fragments per packet: UNIFORM (2,5)
Fragmentation probability: 0.13 (acc. to LBL measurements)
Simulation time: 10 seconds

LOAD (packets/sec.)	50	100	250
PACKET COUNTS (total)			
Input, type MSG	2412	5020	12292
Transmitted, type MSG	2100	4407	10689
Transmitted, type F	1116	2083	5553
Transmitted, total (ALL)	3216	6490	16242
Output, type MSG	2412	5020	12288
% fragment packets	35	32	34
UTILIZATION (%)			
Mean channel utilization	8	16	40
DELAYS			
Mean delay	2.0	2.3	4.4
Max. delay	20	24	44
Stand. dev.	2.8	3.0	4.9
% with delay < 2 msec.	80	74	48
QUEUES (MSG + F)			
Mean input queue length	0.1	0.2	0.8
Max. input queue length	8	13	18
Max. output queue length	6	7	14
MEAN PACKET SIZES (bytes)			
Type MSG	102	102	102
Type F	250	250	250
ALL	150	150	150
ON-THE-FLY STATISTICS			
No of on-the-fly (MSG)	1965	3886	6897
-------"------- (F)	1009	1707	3088
-------"------ (ALL)	2974	5593	9985
% on-the-fly (MSG)	93	88	65
% on-the-fly (F)	90	82	56
% on-the-fly (ALL)	92	86	61

6.3. Token Ring Backbone Simulation

Results are presented in Table 6.3. They display a very "nice" network behaviour.

DELAYS and QUEUES show a very moderate increase as the load increases, and is well within acceptable limits for all loads simulated. Even with 250 packets/sec per gateway throughput, more than 95% of the packets have a delay of less than 2 msec.

The bandwith share between the gateways is fair. The slight difference may be due to statistical differences in generated traffic.

The "Max packets waiting for token" tells the maximum number of gateways waiting for the token during the 10 second simulation time.

6.4. IBX LANmark Backbone Simulation

Simulation results are shown in Table 6.4. Five gateways were simulated to keep simulation time within acceptable limits.

Note that this simulates a case in which each gateway (DOB) has its own 1 Mbps input/output channel. It is technically possible to connect up to 16 gateways to one IBX IM Buffer Board, letting them share the access to the 1 Mbps channel to the IBX LANmark Router.

The packet size distribution for IBX has to be commented on. Refer to the LBL packet size distribution given in Table 6.1. As the maximum packet size for IBX is 266 bytes, one can see that about 13% of the packets will be fragmented. The fragmentation algorithm assumed is that a long packet of length L is divided into N packets of maximum size, and a "tail" packet of length L-(N*266).

For non-fragmented packets (transmitted as type MSG), the service time distribution can easily be deducted from the table in Section 6.1 The fragment size distribution will be different. Since maximum packet length is 1100 bytes, which is greater than 4*266 and less than 5*266, the maximum number of fragments will be 5. The minimum number is of course 2. All but one in a convoy of F-packets will have length 266 bytes. The distribution of the "tail" packet is taken to be uniform in the interval <0,266>. and the number of fragments is taken to be uniform in the interval <2,5>. The packet lengths are converted into transmit times adjusted for a frame overhead of 20 bytes and using a channel speed of 1 Mbps. This yields a service time distribution for F-type packets on the BBI and BBO service nodes as follows:

F: EMPIRICAL(0.25,0.16,2.29,0.75,2.29,2.29)

where 0.25 is the probability that the packet is a "tail" packet, and 2.29 is the transmit time for a 266 byte packet with 20-byte frame overhead. (Note: 0.25 should actually have been 0.28 = 1/3.5, since 3.5 is the average number of fragments).

The input queue lengths are the queues that would build up in a DOB, if such a device was to be used as gateway. It clearly shows that even with a load of 250 packets/sec, and with average packet size of about 175 bytes, the queues will be well within the buffering capacity of a DOB, which is above 100 kbytes in the larger model used for full Ethernet segment gatewaying. The reason is the abundant bandwith available to each of the gateways in the case of only one gateway per 1 Mbps channel.

The output queues are the IBX router queues. To get the total buffering space required for 20 gateways, multiply the numbers given by 4, then multiply by average packet length (175 bytes), giving a maximum buffer space requirement of about 10 k bytes.

The probability for a packet being routed on-the-fly, is almost exactly 1—(channel utilization), as one would expect. The probability for fragments is a little bit less, probably because the fragments make the output channel more busy for a short period of time.

The delays are well within acceptable limits for all simulated loads.

6.5. Comparative Backbone Analysis

Results from the three different simulations are compared in Tables 6.5a-c.

Table 6.5a BACKBONE COMPARISON—BUS/CHANNEL UTILIZATION (%)

LOAD (packets/sec):	50	100	250
Ethernet	15	30	66 *)
Token Ring	14	28	70
IBX LANmark	8	16	40

*) 66 < (2.5*30) due to packet loss

The utilization is, as expected, close to equal for 10 Mbps Ethernet and Token Ring, as long as there is no packet loss for Ethernet. The small difference is due to higher frame overhead for Ethernet. LANmark has more spare capacity left under the present assumptions, where each gateway has a 1Mbps full duplex channel of its own. The available bandwith is in fact twice as large for LANmark (1 Mbps compared to 10 Mbps divided by 20). However, the utilization is more than half that of Ethernet or Token Ring. This is probably due to fragmentation, which adds more frame overhead.

Table 6.5b BACKBONE COMPARISON—DELAYS (msec.)			
LOAD (packets/sec):	50	100	250
MEAN VALUES			
Ethernet	.21	.53	81
Token Ring	.17	.21	.55
IBX	2.0	2.3	4.4
MAXIMUM VALUE			
Ethernet	3.5	107	526
Token Ring	1.7	2.2	13
IBX	20	24	44
STANDARD DEV.			
Ethernet	.30	2.3	84
Token Ring	.20	.25	.74
IBX	2.8	3.0	4.9
% WITH DELAY < 2 msec			
Ethernet	99.6	94.5	16
Token Ring	100	100	95.7
IBX	80	74	48

The delays (Table 6.5b) speaks definitely in favor of the Token Ring. It maintains low delays even at loads where Ethernet has collapsed. Even for light loads, the Token Ring has lower delays, both mean and especially maximum values.

IBX has as expected larger delays than the Token Ring. It is more than 10 times as large for light loads, even though the channel speed is 1/10 of that of a token ring. The reason is that some packets are being stored in the IBX Router, before being output. For heavier loads the ratio is below 10, probably because IBX has greater bandwith pr. gateway, yielding smaller queueing times to compensate for larger transmission times.

Table 6.5c BACKBONE COMPARISON—INPUT QUEUE LENGTHS (packets)			
LOAD (packets/sec):	50	100	250
MEAN			
Ethernet	.001	.01	20
Token Ring	.001	.001	.02
IBX	.1	.2	.8
MAXIMUM			
Ethernet	2	15	51*
Token ring	2	2	5
IBX	8	13	18

*Limited by queue capacity.

In addition to the tabulated information above, remember that Ethernet will lose packets for all loads above 150 packets/sec, mainly due to input queue overflow. For 250 p/s, the loss is as large as 11.5%. None of the other networks have similar type of packet loss.

7. NETWORKS WITH FINITE CAPACITY RECEIVERS

LBL's Computer Science Research Department has previously conducted a very interesting experiment comprising a token ring network with two transmitters and one receiver with finite capacity [5]. The experiment shows among other things that it is possible to get a very unfair reception distribution. Packets from one of the transmitters will be received while all packets from the other will be discarded due to busy receiver at the time the start of the packets arrive at the receiver. The experiment shows that "fairness" will occur only if

$$(n * d) < D < ((n * d) + t) \qquad n = 1,2,...$$

where

- d the transfer time for a packet from main memory to transmitting network interface
- t the packet transmit time on the network
- D the receiver Delay, i.e., the transfer time for one packet from receiving interface into receiver host's main memory, during which the receiver is unable to start to receive another packet.

Model Ring30 is an exact replica of this experiment, and it shows exactly the behaviour we were looking for (Table 7.1). This is taken as part of the validation procedure for the Token Ring model as a whole.

Table 7.1 Token Ring, 2 transmitters/1 receiver.
Receiver distribution for diff. values of D.

D	d = 4.0, t = 1.0	XMTRS	xmt'd	rcv'd	disc'd
0.5	0 < D < t	X1	30	30	0
		X2	30	0	30
2.0	t < D < d	X1	30	30	0
		X2	30	0	30
4.5	d < D < d+t	X1	30	10	20
		X2	30	10	20
6.0	d+t < D < 2d	X1	30	15	15
		X2	30	0	30

Reference [5] also predicts a similar behaviour for Ethernet local networks, and for both types of networks with more than 2 transmitters and more than one receiver. A 2 transmitter/1 receiver simulation was run for Ethernet (model Ether30), and confirms the prediction. Table

7.2 shows results from a 5 transmitter/2 receiver Ethernet simulation, where transmitters X1-3 transmits to receiver R1, transmitters X4-5 transmits to R2. It shows as predicted a fair receiver distribution for $d < D < d + t$. It is unfair for $0 < D < t$, and for $d + t < D < 2d$, in the respect that one of the 5 transmitters never gets a packet to its receiver.

It is left for further study to explain this behaviour and to run more simulations where, for instance, d and/or t is changed so as to avoid or to get collisions on the Ethernet.

Table 7.2 Ethernet, 5 transmitters/2 receivers.
Receiver distribution for diff. values of D.

D	d =4.0, t = 1.0	XMTRS	xmt'd	rcv'd	disc'd
0.5	$0 < D < t$	X1 (R1)	30	29	0
		X2	29	0 !	29
		X3	29	29	0
		X4 (R2)	29	30	0
		X5	29	29	0
4.5	$d < D < d+t$	X1 (R1)	29	6	23
		X2	29	8	21
		X3	29	7	22
		X4 (R2)	30	10	19
		X5	29	9	19
6.0	$d+t < D < 2d$	X1 (R1)	29	0 !	30
		X2	29	10	19
		X3	29	9	20
		X4 (R2)	30	10	19
		X5	29	9	19

8. SUMMARY

The backbone simulations show, or rather confirm, that an Ethernet can hardly act as a backbone interconnecting 20 LANs if the load exceeds 150 packets/sec per gateway. Both IBX and a Token Ring seem to be able to handle at least 250 packets/sec. All networks can at least handle an average of 100 packets/sec.

Note that the IBX simulations are based upon one gateway per IBX Buffer Board. For further comparisons one should consider the cost of this as compared to a Token Ring interface.

The present version of PAWS turned out to be an impressive tool for modelling computer networks. The simulations were set up with a very limited amount of programming effort, using the inherent facilities of PAWS to model different queuing disciplines, different arrival and service distributions, and to get sufficient statistics output. The system seems to be well tested, and no bugs emerged in PAWS during this work. There were some problems in enhancing the system for the largest models, mainly due to lack of information on how to do it, and on which PAWS modules that had to be changed. The TRACE facility should have been more diversified, not only ON/OFF. The ON switch is very abundant on the amount of information generated.

The present models should be a good starting point for further investigations into the behaviour of different LAN's and backbones under different load conditions. For a more complete study, it is highly recommended to include broadband networks and the 80 Mbps Token Ring being developed by Proteon, Inc. It is also feasible to program models of Internets, comprising different types of networks interconnected by gateways.

ACKNOWLEDGMENTS

Thanks to Bob Fink of LBL's Office of Computing Resources and Bill Greiman and Joe Sventek of LBL's Computer Science Research Department for providing the necessary background information, and for valuable discussions throughout. Bob Fink also deserves thanks for translating this from Norwegian-English into English.

This work was supported by the Department of Energy under Contract No. DE-AC03-76SF00098 and by the University of Trondheim, Norway.

AUTHOR ADDRESS INFORMATION

Alf Engdal	Phone:	47-7-592945
RUNIT, Univ. of Trondheim	Telex:	55620 sintf n
Strindvn. 2	ARPA:	alf@lbl-csam
N-7034 Trondheim-NTH		alf@nta-vax
NORWAY	QZCOM:	Alf_Engdal

REFERENCES

[1] Metcalfe, R.M and D.R.Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks," *Communications of the ACM*, Vol. 19, July 1976, pp. 395–404.

[2] Tobagi, F. and V.B. Hunt,"Performance Analysis of Carrier Sense Multiple Access with Collision Detection," *Computer Networks*, No 4. 1980, pp. 245–259.

[3] Shoch, J.F. and J.A. Hupp, "Measured Performance of an Ethernet Local Network," *Communications of the ACM*, Vol. 23, December 1980, pp. 711–721

[4] Meisner, N.B., "Estimating the throughput of broadband LANS," *Systems and Software*, Nov. 1982, pp. 75–77.

[5] Sventek, J., W. Greiman, M. O'Dell, A. Jansen, "Token Ring Local Area Networks. A Comparison of Experimental and Theoretical Performance," Computer Science and Research Department, Lawrence Berkeley Laboratory. DRAFT 1983.

[6] Draft IEEE Standard 802.3, CSMA/CD Access Method and Physical Layer Specifications, Rev. D, Dec. 1982.

[7] IBX S/80 Integrated Business Exchange. General Description. InteCom, Inc., Feb. 1983.

[8] PAWS 2.0 Performance Analysts Workbench System. User's Manual. December 2, 1982. Information Research Associates.

[9] Operation and Maintenance Manual for the Pronet Local Network Interface. Proteon Associates, Inc., November 1981.

[10] The Token Ring Draft Standard. IEEE 802.5. May 21, 1984.

This report was done with support from the Department of Energy. Any conclusions or opinions expressed in this report represent solely those of the author(s) and not necessarily those of The Regents of the University of California, the Lawrence Berkeley Laboratory or the Department of Energy.

Reference to a company or product name does not imply approval or recommendation of the product by the University of California or the U.S. Department of Energy to the exclusion of others that may be suitable.

TECHNICAL INFORMATION DEPARTMENT
LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720