INDEX TERMS:

Local Area Networks, Ethernet, collision protocols, two-cable Ethernet, segmented channels, performance, fairness.

ADDRESS FOR CORRESPONDENCE:

Wlodek Dobosiewicz

University of Alberta

Department of Computing Science

615 General Services Building

Edmonton, Alberta T6G 2H1

CANADA

FOOTNOTES:

[1] This work was supported in part by NSERC grants A9110 and A9183.

[2] Technically, the jam may be just a copy of the information inserted into the recipient's part of the bus.

[3] The jamming signal may be just a copy of the packet information being transmitted.

[4] Note that shifting the length of all inter-packet spaces by a constant does not affect the fairness.

[5] Due to the imprecise notion of time at different stations, *A* and *B* never use inter-packet spaces of exactly the same length.

[6] Equation 7.1.5 can be written in a simpler form by considering only stations *S* and *C* (note that the terms independent of *S* contribute to a constant in η_t. The present form of equation 7.1.5 is left unsimplified for compatibility with equations 6.9 and 7.2.1.

[7] The inter-packet spaces were slightly randomised – to simulate the imperfect notion of time at particular stations.

[8] One can think of some modifications to the *SCS* protocol that would enable stations to determine whether the silence period follows one transfer or two simultaneous transfers. The authors have actually implemented these modifications and calculated the corresponding η function which turned out to be rather complicated and obscure. No noticeable improvement in the network behaviour was observed, however.

FIGURE CAPTIONS:

Fig. 1. The SCS network.

Fig. 2. Collisions in the SCS network.

Fig. 3. The DCS network.

Fig. 4. Performance characteristic of SCS.

Fig. 5. Performance characteristic of DCS.

Fig. 6. Performance characteristic of SCS (biased traffic).

Fig. 7. Performance characteristic of DCS (biased traffic).

Fig. 8. Message delay distribution for the SCS network.

Fig. 9. Message delay distribution for the DCS network.

Fig. 10. Message delay distribution for the DCS network with packet spacing adjustment.

Fig. 11. Message delay distribution for Ethernet.

Fig. 12. Message delay distribution for Ethernet with packet spacing adjustment.

Fig. 13. Message delay distribution for the SCS network with packet spacing adjustment.

ON TWO MODIFIED ETHERNETS[1]

Wlodzimierz Dobosiewicz   Pawel Gburzynski

University of Alberta

Department of Computing Science

ABSTRACT:

The paper presents two variations of a Local Area Network based on a modified concept of Ethernet. Unlike many other concepts (e.g. collision-free protocols) the proposed modifications are simple, realistic and inexpensive. In particular, the essence of the CSMA-CD protocol is retained. The performance of the new networks is demonstrated to be superior to the performance of standard Ethernet. The second network (DCS) seems to be particularly interesting, as (despite the collision protocol) its asymptotic effective throughput turns out to be equal to the theoretical maximum. The presented networks are used to demonstrate a certain approach to investigating and eliminating the unfairness of a bus protocol in which stations can transmit spontaneously.

1. Introduction

A bus-type Local Area Network (e.g. Ethernet) is a competitive alternative to a linked structure – in which particular stations are directly interconnected via dedicated links. The latter approach to the local networking suffers from some painful disadvantages. If the stations are not fully interconnected (a separate link for each pair of stations), we have to cope with the problems of packet routing and accept the possibility that a failure in some of the stations may disconnect the network. On the other hand, fully connected topologies are only acceptable to a very limited extent.

The main advantage of an Ethernet-type bus lies in the fact that there is only one channel whose nature is indifferent to the number of stations forming the network. As long as the physical structure of that channel is simple and the access protocol is not too complicated, the simplicity, reliability and flexibility of that type of network cannot easily be matched by the competition.

However, there are several reasons why the competition has not been eliminated ultimately. Essentially, they can be summarised in the following two points:

1. In a bus network, only one station can transmit a packet at a time. The entire channel must be reserved for the duration of a single transfer.

2. The bus access protocol (CSMA-CD) tends to waste a part of the bus transmission capacity under heavy traffic conditions (to resolve collisions between stations that attempt to acquire the bus at approximately the same time).

At first sight, the first problem cannot be alleviated: if the medium is shared by all stations then, unquestionably, it must be acquired on an exclusive access basis. The second problem can be fought by trying to employ better protocols.

The essence of the CSMA-CD protocol (*Carrier Sense Multiple Access with Collision Detection*) is in the stations' ability to sense the carrier prior to and during a transfer attempt and to detect collisions caused by multiple packets being transmitted at the same time. A station detecting a collision stops its transfer immediately and backs-off (reschedules the retransmission of its packet after a random delay). With this solution, a collision always wastes a part of the bus capacity: some information inserted into the bus is lost.

Several attempts were made to define collision-free protocols applicable to an Ethernet-type network (e.g. BRAM-type solutions – cf. [1] and [24], or Token Bus - cf. [23]), or to devise a collision protocol of a superior quality (cf. [5,7,9,12,17]). The proposed solutions were either unrealistic (e.g. assuming common notion of time for all stations) or expensive and unreliable (e.g. sophisticated techniques of monitoring bus activities and maintaining dynamic priorities at the stations). Practically, only the idea of Token Bus has been publicly accepted so far and turned into a standard [23]. However, the performance of Token Bus, being very good for heavy loads of the network, is definitely inferior to the performance of Ethernet under low and moderate traffic conditions. This is due to the substantial overhead caused by passing an explicit token packet.

A simple way to improve the performance of the CSMA-CD protocol is in enhancing the quality of the backoff method which determines the mean retransmission delay after a collision. Although some (rather marginal) progress has been made in this area since the Binary Exponential Backoff was originally employed in Ethernet (cf. [3,15,18,19]), a further improvement seems very unlikely.

Our work, whose results are presented in this paper, has been aimed at improving the performance of an Ethernet-type network. However, instead of trying to devise a better bus access protocol, we attempt to approach the problem from the other end. The idea is to relax the assumption about the connectivity of the bus. In our proposal, the bus does not behave like pure *ether* any longer. However, the resulting networks still remain much closer to an Ethernet-type network than to a linked structure. In particular, the following properties of Ethernet are fully retained:

1. The physical nature of the bus remains (almost) as simple as in Ethernet (e.g. a coaxial cable). The medium is bidirectional, the stations can be active and passive: in the latter case, the power budget, critical in the case of fibre networks [14], is not a consideration.
2. The bus remains the exclusive information passing medium. Stations need not relay any packets in order to make them reach their destinations.
3. If there is only one station willing to send a packet, the packet is sent and received at the destination with the full transmission capacity of the medium. Essentially, stations use a regular CSMA-CD protocol which is costless as long as there are no collisions.
4. As long as the bus is physically reliable, the sender of a packet is able to determine whether the packet has reached the destination in a "good shape".

In their original versions, the proposed protocols are inherently unfair in the sense that not all stations have equal chances for transmitting their packets successfully. In this paper we also investigate these problems and show how they can be eliminated.

The two networks discussed in this paper have been implemented as modifications to the commercially available Ethernet hardware (cf. [16,21,22]). The implementation was done in LANSF (cf. [4,5,6]) − a software system for emulating communication protocols. LANSF has been designed and implemented by one of the present authors and is available upon request.

2. The SCS Network

We assume that the network consists of a number of stations distributed along the bus. For convenience, the two ends of the bus will be referred to as the **left** end and the **right** end. The length of the bus is expressed as the **end-to-end propagation delay** (L) in bit-slots (e.g. for a 10Mb/s Ethernet, 1 bit-slot = 100*ns*). Each station is assigned a specific location on the bus. The location of a station is represented as its distance (in bit-slot time units) from the left end of the bus.

Every station is equipped with a switch allowing it to disconnect the channel (see fig. 1). A station willing to send a packet must know the side of the bus (with respect to itself) on which the recipient of the packet is located.

The simpler of the two proposed networks (*SCS* - **Single Channel with Segmentation**) operates similarly to regular Ethernet, according to the following rules:

(SCS-1) When a station attempting to send a packet senses that the bus is idle, it keeps on sensing for a specified period of time (**packet spacing**) and then, provided that the bus is still idle, it disconnects the channel and starts to transmit the packet. The packet is inserted into the part of the cable that corresponds to the location of the recipient. While sending the packet, the station inserts a jamming signal into the other (unused) part of the cable. Immediately after the transfer is completed (or aborted) the channel is connected back.

(SCS-2) If a transmitting station senses interference (some other activity) on the recipient's side of the bus, then it aborts the transfer, reconnects the channel and assumes that a collision has occurred. Next, the station sends a jamming signal (into both parts of the channel) to make sure that any other party involved in the collision also recognises it. Then the stations reschedules its transfer attempt according to the

Binary Exponential Backoff algorithm.

One should expect that the proposed modifications will reduce the number of collisions by ignoring some of the activities that could potentially interfere with a packet being transmitted. Whatever happens in the unused (from the viewpoint of the current transfer) part of the channel will not interfere with that transfer. However, due to the the jam[2] inserted by the sender into the unused part of the bus, some of the activities (transfer attempts) started by the stations located in that part may be garbled and destroyed. The point is that the sending station must indicate the fact that the channel has been disconnected (by forcing collisions for the transfers attempting to cross the separation point). This is necessary because the sender of a packet must be able to determine whether the packet has reached its proper destination. Disconnection can be implemented (by an electronic switch) to operate in a small fraction of one bit-slot time.

The modified network acquires (to a certain extent) some of the positive properties of a linked network. The disconnected channel behaves as two (or more) separate links and, although the interference between those links is not eliminated totally, the chances for a collision are reduced. Let us assume that two randomly selected stations *A* and *B* attempt to transmit their packets at approximately the same time (fig. 2). Without loss of generality we may assume that *A* is closer to the left end of the bus than *B*. In regular Ethernet the two packets would collide and the information in both packets would be lost. Now, there are four possibilities:

1. *A* transmits to the right, *B* transmits to the left.

   In this case both packets collide and are destroyed (both transfers go into the same segment of the bus).

2. A transmits to the left, B transmits to the right.

   No collision. The two transfers are in the opposite directions and go into separate segments. Both packets reach their destinations.

3. Both A and B transmit to the right.

   The packet sent by B reaches its destination. However, the packet sent by A collides with the jam emitted by B.

4. Both A and B transmit to the left.

   This situation is analogous to 3. The packet sent by A reaches its destination. The B's packet collides with the jam emitted by A.

Let us calculate the expected number of packets passing through if two stations attempt to transmit their packets at the same time. Assume that the bus is represented as the [0,1] interval densely and uniformly populated by stations. Let A and B be two points (stations) arbitrarily chosen from [0,1], such that $A \le B$. The probabilities of the four cases listed above are respectively:

1. $(1-A)\times B$
2. $(1-B)\times A$
3. $(1-B)\times(1-A)$
4. $A\times B$

No packet is successfully transmitted in case 1. For the remaining cases, the numbers of successful transfers are 2, 1, and 1, respectively. Therefore the expected number of successfully transmitted packets is:

$$E\,(A,\,B) \;=\; 1-B+A \qquad (A \le B) \tag{2.1}$$

Symmetrically, when $A > B$, this expectancy is:

$$E\,(A,\,B) \;=\; 1-A+B \qquad (A > B) \tag{2.2}$$

Therefore, irrespective of the mutual position of A and b, the expected number of packets transmitted successfully is:

$$E\,(A,\,B) \;=\; 1 \;-\; \left|B - A\right| \tag{2.3}$$

The overall expectancy for all possible combinations of A and B is given by:

$$\int_0^1 \int_0^1 1 - \left|B - A\right| \; dA\; dB \;\; = \;\; \frac{2}{3} \tag{2.4}$$

The above formula would suggest that the expected number of packets destroyed due to collisions should be reduced by one third as for two colliding stations, $2/3$ of a packet (on the average) are successfully transmitted. Actually, the situation is not that simple. First, in the case when there are more than two contending stations, only the leftmost and the rightmost stations can win. The expected number of successfully transmitted packets is then equal to $2/(k+1)$, where k is the number of colliding stations. Second, reducing the number of fatalities from collisions helps reduce the overall number of collisions, as the non-destroyed packets are not retransmitted and, in consequence, do not increase the level of the incoming traffic.

Unfortunately, the modified network also carries over some of the negative properties of a linked network. One of them appears as a simple variant of the routing problems. Namely, each station is supposed to know the side of the bus on which the recipient of the packet is located. This problem does not seem to be very serious, however. It is enough to augment the local address table at each station with one extra bit for each entry. Although the network loses some of its flexibility (changing locations of some stations may require some update of the local address tables), it only occurs when something changes in the network. In reality, changing locations of some stations without adding anything to the network (which requires updating of the local address tables anyway) doesn't seem to happen very often.

One could claim that separating the bus with switches at each station affects the reliability of the network. However, a switch is a fairly simple device and it is only triggered by a transfer attempt. Of course it is still theoretically possible that a failure in a station could disconnect the bus (e.g. in consequence of an infinite transfer). Let us note that similar failures would block regular Ethernet as well.

3. The DCS Network

Although it is clear that in the *SCS* network presented above the number of packets destroyed due to collisions tends to be substantially lower than in a regular Ethernet, the bus is still not utilised at its full capacity. Namely, it may happen that a collision is fatal for all the parties involved (cf. case 1 above). One could observe that the bus utilisation occasionally jumps over 100% (case 2). Alas, due to the unavoidable penalty associated with a fatal collision (backoff, collision jam, packet spacing), those cases do not fully compensate for fatal collisions.

Let us now imagine a network with the segmentation facility described above whose bus consists of two separate cables (fig. 3). We will call it a *DCS* network (**Dual Channel with Segmentation**). Each station is connected to both channels (cables) and has a disconnecting switch on each cable. Conceptually, traffic on each channel is unidirectional, although jamming signals are propagated in the opposite direction than packets. Whenever a station wants to send a packet to the left it uses one channel (say cable 1). The other channel (cable 2) is used for transfers in the opposite direction. Each station has to listen for packets addressed to itself on both channels as it may happen that two packets addressed to one station arrive at (approximately) the same time (from the opposite directions). In our experiments (their results are presented in next sections) we have assumed that each station is able to receive packets on both cables at the same time, while only one transfer attempt can be active at a station at a given moment. This way only a part of the station's logic has to be doubled (with respect to the network described previously). Note that at least two other solutions are possible:

In a more costly implementation, a station might be allowed to send two packets simultaneously (if they happened to go in the opposite directions).

In a cheaper version, a station would not be able to receive two packets at a time. Thus, a station receiving a packet on one cable and detecting another packet addressed to itself on the second cable would send a jam (force a collision) to destroy the second packet. The sender of the second packet would then detect the collision, abort the transfer and reschedule it at a later time. This way we would avoid doubling the receiver's buffer space and fulfill the postulate that the sender of a packet be able to tell whether the recipient has received it. The loss in performance (at least for the uniform traffic) seems to be negligible, as the chances that a station would receive two packets at the same time are not very big. Should the traffic be biased, we could implement the biased station(s) differently – doubling their buffer space and allowing them to receive packets on both cables simultaneously.

With the *DCS* network, the transfer protocol has to be modified as follows:

(DCS-1) Before a transfer attempt, the station determines which channel is to be used for the transfer (according to the relative location of the recipient of the packet). The bus is assumed to be idle if there is no activity on the selected channel.

(DCS-2) After the bus is "sensed idle" and after obeying the packet spacing rules, the station disconnects the selected channel and uses its appropriate segment to send the packet. The other segment of the channel is jammed[3] at the same time.

(DCS-3) During the transfer attempt, the sending station listens for a collision (another activity) only on the recipient's side of the disconnected channel. As soon as a collision is detected the transfer is aborted. **No collision jam is sent**. The transfer attempt is rescheduled at a later time.

Only one collision type is now possible: a transfer attempt may only collide with the jam from another transfer attempt. Basically, there is only one party fatally involved in such a collision. Therefore, no collision jam is necessary as there is no other party that has to recognise the failure of its transfer attempt. This collision mechanism resembles the operation of LCSMA-CD of Maxemchuk [14]. The reader may notice a superficial similarity of DCS to DQDB [10]; a more careful inspection reveals, however, that while DQDB uses feedback-based reservation system, DCS is based on collision-arbitrated access control and uses neither slots nor head stations.

Several transfer attempts may be active on each channel at any given moment. All of them except one will collide fatally and will have to be rescheduled. However, the leftmost attempt for the left-directed channel and the rightmost attempt for the right-directed channel will succeed. Therefore, when the network load approaches the saturation point (most stations have ready packets to be transmitted) the utilisation of each channel approaches 100% of its theoretical effective throughput. Indeed, let us assume that the network consists of infinitely many stations equally distributed along the bus and that the traffic is uniform. Then the following fact holds:

Let k be the number of stations willing to transmit their packets at the same time and $E(k)$ be the expected number of successful transmissions. Then

$$\lim_{k \to \infty} E(k) = 2 \tag{3.1}$$

Proof

It is clear that if at least two of the k transfer attempts are in the opposite directions, exactly two transmissions will be successful. Thus the expected number of successful attempts is given by:

$$E(k) = 2 - P_l(k) - P_r(k) \tag{3.2}$$

where $P_r(k)$ and $P_l(k)$ are the probabilities that all k transfer attempts go to the left and to the right, respectively. By the simple rules of symmetry $P_l(k) = P_r(k)$ and thus:

$$E(k) = 2 - 2P_l(k) \tag{3.3}$$

To calculate $P_l(k)$ let us assume (see section 2) that the bus is transformed into interval [0,1] densely and uniformly populated by stations (represented by points). If the traffic is uniform, the probability that the packet transmitted by a station S goes to the left of S is equal S (where S is interpreted as a point from [0,1]). Thus the probability that all k randomly chosen stations $S_1, \ldots, S_k$ transmit their packets to the left is determined as:

$$P_l(k) = \int_{I_k} S_1 \times S_2 \times \cdots \times S_k \; dS_1 dS_2 \; \cdots \; dS_k \tag{3.4}$$

where I_k is the k-dimensional unit cube. Consequently, we have:

$$\int_{I_k} S_1 \times \cdots \times S_k \; dS_1 \; \cdots \; dS_k = \int_{I_{k-1}} S_1 \times \cdots \times S_{k-1} \left[\frac{1}{2} S_k^2 \right]_0^1 dS_1 \; \cdots \; dS_{k-1} = \tag{3.5}$$

$$\frac{1}{2} \int_{I_{k-1}} S_1 \times \cdots \times S_{k-1} \; dS_1 \; \cdots \; dS_{k-1}$$

Thus, taking into account that $P_l(1) = 1/2$, we get:

$$P_l(k) = \frac{1}{2} P_l(k-1) = \left[\frac{1}{2} \right]^k \tag{3.6}$$

and finally:

$$E(k) = 2 - \left[\frac{1}{2} \right]^{k-1} \tag{3.7}$$

□

Besides demonstrating that the asymptotic throughput of *DCS* is actually equal to the theoretical maximum for each the two channels, the above fact also shows that the throughput improves when more stations become active. Let us note that for the *SCS* network (and also for Ethernet) the throughput decreases to 0 as the number of stations attempting to transfer their packets at the same time grows. The property of *DCS* shown above is usually attributed to protocols with explicit reservation of the channel, e.g. token protocols. Unfortunately, although DCS uses 100% channel capacity, it is not a **true** capacity-1 protocol for *Big-a* networks [13], since the minimum packet length is a linear function of the network size. To a certain sense, the same can be said of FDDI [11] where the maximum throughput achieved by the network is a function of *TTRT*.

At first sight it would seem that the *DCS* protocol need not use any backoff function, i.e., the retransmission delay after a collision should be 0. Indeed, collisions for *DCS* are not fatal: there always is a winner. However, one unpleasant property of the network, in its version presented above, is that not all stations have equal chances for sending their packets successfully. If two stations attempt to send packets to the left at approximately the same time, then the station located more to the right will most likely lose. In other words, the transfer protocol imposes some implicit (and not always desirable) priorities on the stations. In section 5 we show how to cope with this problem. For now we can assume that the backoff algorithm borrowed from Ethernet is used as an "equalising agent" which delays randomly multiple contending stations, so that they do not attempt to transmit all at the same time.

Let us note that broadcast-type messages can easily be incorporated into each of the two proposed networks. A station sending a broadcast message would not disconnect the channel. However, introducing broadcast messages into *DCS* would result in a potential for fatal collisions (in which all colliding packets are destroyed). Therefore, the collision jams and the randomised backof algorithm should be retained in that case.

If the need arises, DCS can be augmented with **erasure nodes** [2,20] in which case the results will be the same as when erasure nodes are added to other protocols, e.g., DQDB.

4. The Average Performance of SCS and DCS

Adequate mathematical models of the two networks proposed above do not seem to be tractable. The non-symmetric contribution of particular stations to the network traffic is the most serious problem. Therefore, the behaviour of the two networks has been analysed by emulation. The networks and the protocols were programmed in LANSF. LANSF (the Local Area Network Simulation Facility) is a system for modelling networks and distributed systems developed by one of the authors at the University of Alberta (cf. [4,5,6]). The main advantage of LANSF is in the accurate description of the real network. The source code of the system (in C) together with complete documentation is available from the authors upon request.

The network models were exact, in the sense that all the relevant elements of the real networks were carefully reflected in the models. In particular, the notions of distance between stations, packet spacing and collision jams (where applicable) were implemented.

The parameters of the investigated networks were set up as for a 10Mb cable (for the two-cable bus it means 10Mb in each direction) with 50 stations distributed uniformly along the bus. The new networks were implemented as "minimal" modifications to the standard Ethernet – to give a realistic basis for meaningful comparison. The parameters of the reference Ethernet corresponded strictly to the commercial network and were as follows (cf. [16,22]):

Minimum packet size	368 bits;
Maximum packet size	12000 bits;
Frame information	208 bits;
Minimum packet spacing	96 bits;
Minimum jam size	32 bits;

Bus length 50 bits.

The experiments were carried out for various bus lengths and numbers of stations. The result reported here were obtained for the bus length of 50 bits and 50 stations equally spaced along the bus. The results for other cable lengths and/or numbers of stations were qualitatively consistent with those presented in this paper. For a longer cable and/or more stations they were even more in favour of the proposed networks. Each single point of a performance curve (fig. 4-13) was obtained by observing the network behaviour for the amount of time required to pass through it about 200,000 messages.

Two traffic patterns were examined. In the *uniform pattern* the average amount of traffic generated at each station (and addressed to each station) was the same. In the *biased pattern* 97% of all messages were either sent by or addressed to (with equal probability) one distinguished station. In both cases the arrival of messages formed a Poisson process. The mean interarrival time was a simulation parameter (to cover the variety of traffic conditions). The message length was exponentially distributed with mean value of 1000 bits (125 bytes).

The performance of a network is usually described as the correlation of the throughput and delay. In our case we used the following versions of these measures:

The so-called **effective throughput** (denoted by h_e) was calculated as the number of message bits (header and trailer excluded) successfully received at destinations, divided by the simulation time (expressed in bits).

The **relative message delay** (denoted by d_m) was measured as the average delay of a message bit counted from the time the message was queued at the sender to the time the packet containing that bit was completely received at the destination.

Figure 4 compares the performance of *SCS* and Ethernet for the uniform traffic pattern. The relative gain in the maximum throughput is about 13%.

When analysing the properties of *DCS* we have to admit that comparing it directly to Ethernet would not make much sense. It is obvious that *DCS* should perform much better than regular single-cable Ethernet as the traffic is now split into two separate streams, each stream being transmitted at least with the capacity of one standard Ethernet. Therefore, the performance of *DCS* is related to the performance of the following networks:

1. Ethernet×2

Ethernet with a two times faster medium (the throughput is multiplied by 2, the delay is divided by 2).

2. SCS×2

The *SCS* network with a two times faster carrier.

3. 2-channel Ethernet-I

A dual-channel Ethernet (without segmentation) in which each station is connected to two cables. Whenever a station wants to transmit a packet it picks one cable at random. If the cable is sensed idle it is used for the transfer, otherwise the other cable is examined. If both cables are busy the station waits for whichever cable becomes idle first, and then uses it to send the packet.

4. 2-channel Ethernet-II

Another two-cable Ethernet, similar to Ethernet-I, in which each cable is used (by convention) for transfers in one direction only. If a station wants to send a packet it determines the direction of the recipient and then uses the cable dedicated for transfers in that direction (following the regular Ethernet protocol).

The performance of the *DCS* network turns out to be superior to the performance of all four reference networks (fig. 5). Although for the low and moderate traffic conditions *Ethernet*×*2* and *SCS*×*2* perform better than *DCS* (for obvious reasons), the maximum throughput of *DCS* is about 16% higher than for 2-channel Ethernet-I, about 23% higher than for 2-channel Ethernet-II and for *Ethernet*×*2*, and about 9% higher than for *SCS*×*2*.

The simulation conducted for the biased traffic pattern produced results which were very similar to those obtained for the uniform traffic. Two series of experiments have been carried out. First, the biased station was located in the middle of the bus (the optimistic case). Next, the biased station was moved to one (left) end of the bus (the pessimistic case). The results for the biased station located in the middle of the bus are presented in fig. 6 and 7. The other results were very similar as far as the relative improvement in the performance was concerned.

5. The Observed Unfairness of SCS and DCS

As we have already mentioned in section 3, the segmentation protocol used in *SCS* and *DCS* imposes some priorities on the stations. A station located close to the left end of the bus and sending a packet to the left has a higher priority than, say, the rightmost station sending a packet in the same direction. One could argue that on the average those priorities are equal (at least for the uniform traffic pattern and the uniform distribution of stations along the bus) as the poor chances for a successful transfer in one direction are compensated by better chances for transfers in the opposite direction. However, the compensation is not full and the networks are not fair to all stations.

We assume here the following natural empirical definition of fairness: "a network is fair, iff the observed average message delay measured under the uniform traffic conditions does not depend on the transmitter's location on the bus". Note that we could also postulate that the message delay be independent on the receiver's location. However, the unfairness of the networks discussed here manifests itself for relatively heavy traffic conditions. Under these conditions, the impact of the sender-receiver distance on the message delay becomes negligible.

Figure 8 depicts the difference in the message delay measured for *SCS* transmitters connected to different places on the cable. The *x*-axis represents the distance in bits of the transmitter from the center of the bus (l_c). The *y*-axis shows the ratio of the message delay measured at the particular distance from the center of the cable (d_l) to the average message delay measured globally for all transmitters (d_{avg}). The five curves on the figure correspond to different network loads; the numbers associated with the curves denote mean message inter-arrival times (in bits).

When the traffic becomes heavy, the difference between the stations located at the ends of the bus and those located close to the middle of the cable becomes quite significant. Even for a relatively moderate mean inter-arrival time of 2000 bits the location-implied priorities of transmitters are clearly visible. For the inter-arrival time of 1600 bits, the message delay observed for a transmitter located at the end of the cable is over 800 times bigger than for a transmitter located in the middle of the bus.

The influence of the station's location on its chances for a successful transfer is even more serious for *DCS* (figure 9). For very heavy load we observe a strange behaviour of the network, namely the location-implied bias seems smaller than for a lighter traffic. This is due to the fact that the stations located at the ends of the bus are starved. Their packets simply do not get through and in consequence the message delay (which is calculated only for those packets that reach their destinations) is no longer a valid measure of the station's priority.

The inter-arrival times on figure 9 are different from those used on the previous figure for two reasons:

(a) *DCS* has two cables instead of one. These times should be multiplied by 2 to become comparable with *SCS*.

(b) The performance of *DCS* (measured separately for each of its channels) is superior to the performance of the other network (see section 4).

6. Eliminating the Unfairness of DCS

In this section we discuss the problems of location-dependent priorities in the *DCS* network. It turns out that although for *DCS* these problems are more serious than for *SCS*, they can be solved in a nicer and more general way.

Let us note that the problem of starvation in *DCS* is of a totally different nature that the problem of potential statistical starvation in Ethernet. Although in Ethernet it is theoretically possible that due to an unfortunate configuration of events some stations (packets) are delayed indefinitely, the probability of that happening converges to zero as the delay grows (cf. [8]). In the *DCS* network, its is possible that a station having a continuous supply of messages going into one side of the bus starves (deterministically) all the stations located on the opposite side.

It is obvious that if starvation is to be eliminated, a station should not be allowed to transmit its next packet immediately after completion of the previous transfer, especially if the two packets go in the same direction. The station should delay its next transfer attempt by a time amount sufficient to give the other stations, located on the side of the bus opposite to the direction of the transfer, a chance to capture the channel. That delay must be randomised, so that we are talking about chances – not certainty which would potentially lead to a starvation again.

A natural solution is to introduce a location-dependent scheme of inter-packet spacing in which the length of the inter-packet space is randomised, and its mean value depends on the location of the transmitter on the bus and on the direction of the transfer. Instead of specifying the conditions under which our network is absolutely fair (which at this moment seems to be intractable), we will try to find some necessary condition – which must be satisfied by a fair network. This condition will help us direct our experimental search for a method that could be used by a station to determine its inter-packet space in such a way that the observed behaviour of the network is "fair".

Without loss of generality, we can assume that there are only two stations *A* and *B* willing to transmit their packets in the same direction. If there are more stations, it is enough to discuss the competition between each pair. If the competition is fair for each pair, then it is fair for all the contending stations.

The two stations *A* and *B* are monitoring the bus and waiting for the beginning of a silence period – to start counting their inter-packet spaces. With the exception of initialisation, the beginning of a silence period sensed by a station in the *DCS* network is always due to the completion of a successful transfer. Thus, there is a station *C* that completes its transmission. The silence period following the end of that transmission propagates along the bus and, in due time, it reaches the stations *A* and *B*. Having sensed the beginning of the silence period, each of the two stations obeys the inter-packet spacing rules and, provided that the channel is still idle, starts transmitting its packet. Let us assume that the two stations want to transmit to the left and that *A* is located on the left of *B*. If no other stations interfere, the following two scenarios are possible:

(1) *A* starts its transfer attempt before the transfer initiated by *B* reaches *A*. Thus *A* wins and *B* aborts its transfer after sensing the jamming signal inserted by *A*.

(2) Before A decides to start its transfer, it senses the transfer initiated by B. Thus A is preempted by B, and B is the winner.

We say that the competition between A and B is fair, if on the average the two stations have equal chances, i.e., each of the two stations wins half of the time.

Let us introduce the following notation:

π the minimum inter-packet space that must be obeyed by all stations in the network, irrespective of their locations;

$d(S_1, S_2)$ the propagation distance between stations S_1 and S_2;

$l(S)$ the location of station S, i.e., the propagation distance from the left end of the cable to S;

By $\sigma(S)$ we will denote the length of the inter-packet space used by S for transfers to the left. Note that transfers going in the opposite directions do not compete and can be considered separately. Obviously, the two directions are totally symmetric and all results obtained for the transfers to the left are directly transformable into the corresponding results for the transfers to the right. From now on we will consider transfers to the left only. The actual inter-packet space used by a station S transmitting its packet to the left is given by:

$$\Pi(S) = \pi + \sigma(S) \tag{6.1}$$

Let us now return to the three-station scenario discussed above. Let t_s be the time moment when C completes its transmission. Stations A and B will start their transfer attempts at:

$$t_A = t_s + d(A, C) + \Pi(A) \qquad \text{and} \tag{6.2}$$

$$t_B = t_s + d(B, C) + \Pi(B)$$

respectively. We would say that their chances for a successful transfer are equal if:

$$E(t_A - t_B + l(A) - l(B)) = 0 \tag{6.3}$$

where E is the expectancy and $l(S)$ is the location of S on the bus. Such a situation is called **strong fairness** of the network. The above equation, which says that each of the stations A and B has equal chances of preempting the other station, can be rewritten as:

$$E(\sigma(A) - \sigma(B) + d(A, C) - d(B, C) + l(A) - l(B)) = 0 \qquad (6.4)$$

Unfortunately, it is impossible to fulfill equality 6.4 for all configurations of three stations A, B, and C. Although it is theoretically possible that stations A and B learn the identity (and thus the location) of C by reading the header of its packet, neither A nor B can know the identity of the other station. Therefore, we have no means to enforce strong fairness: the location-specific packet spacing scheme that we would like to implement to equalise stations' priorities must be based on some average measures and observations. Thus we will consider **weak fairness** and postulate that equality 6.4 hold "on the average", i.e., for each station A, with the expectancy taken over all potential stations B and C.

Let us consider the general case of a *DCS* network consisting of n stations $S_1,..,S_n$ and servicing a given traffic pattern. The traffic pattern associates with each station S_i the probability F_i of this station being used as a transmitter. Let $\eta(S) = E(\sigma(S))$. The postulate expressed at the end of the previous paragraph boils down to the following equation:

$$\sum_{j=1}^{n} F_j \times \sum_{k=1}^{n} \left\{ F_k \times (\eta(S_i) - \eta(S_j) + d(S_i, S_k) - d(S_j, S_k) + l(S_i) - l(S_j)) \right\} = 0 \qquad (6.5)$$

Equation 6.5, being actually a system of n equations parametrised by i, gives an implicit definition of $\eta(S_i)$, $i = 1,..,n$. Although the adequate distribution of σ cannot be determined from 6.5, it seems that the above equation is a very important necessary condition describing a fair *DCS* network. The above system of equations can be solved for every given (realistic) network configuration and traffic pattern.

Note that the *DCS* network employing the packet spacing method outlined above need not employ any after-collision backoffs. The colliding stations that have lost the contention can just reattempt their transfers after sensing the bus idle and obeying the (properly randomised) inter-packet space. In *DCS*, there always is a winner from a collision. The role of the backoff function (the fair resolution of contention) is played by the properly randomised inter-packet spaces.

Let us now try to determine how $\eta(S)$ depends on the location of S for the idealistic network considered in section 3. Thus we will view the bus as the [0,1] interval uniformly and densely populated by stations.

Let $S \in [0,1]$ be an arbitrary station. Equation 6.5 for S takes the following form:

$$\int_0^1 \Xi(B) \times \int_0^1 \Xi(C) \times \left\{ \eta(S) - \eta(B) + \left| S - C \right| - \left| B - C \right| + S - B \right\} dC dB = 0 \qquad (6.6)$$

where Ξ is the density function describing the distribution of transmitters along the bus. For the uniform traffic pattern, the probability that a randomly chosen packet addressed towards the left end of the bus is transmitted by a station located to the left of S is equal to S^2. Thus $\Xi(S) = 2S$ and the above equation turns into:

$$\int_0^1 B \times \int_0^1 C \times \left\{ \eta(S) - \eta(B) + \left| S - C \right| - \left| B - C \right| + S - B \right\} dC dB = 0 \qquad (6.7)$$

By simple calculations we determine that equation 6.7 is satisfied by the function:

$$\eta(S) = \delta - \frac{2}{3} \times S^3 \qquad (6.8)$$

where δ is a constant. We are free to assume any value of δ, as long as $\eta(S)$ is non-negative for every S. It is reasonable to put $\delta = 2/3$ which is the minimum value of δ with this property[4]. The minimum δ that makes η non-negative is $2/3$, thus we finally have:

$$\eta(S) = \frac{2}{3}(1 - S^3) \qquad (6.9)$$

Equations 6.5 and 6.9 can be used to determine the mean length of the inter-packet space used by a station. However, they say nothing about the distribution from which the actual values of that length are to be drawn. The most natural suggestion is to use the uniform distribution, i.e., postulate that the actual length of the inter-packet space be described by the following formula:

$$\sigma(S) = U(0, 2\eta(l(S)/L) \times L) \quad (6.10)$$

where L is the end-to-end propagation delay of the cable and U denotes the uniform distribution within the specified interval.

Experiments conducted for the reference network with σ determined by equations 6.9 and 6.10 showed that stations located closer to the center of the bus were still slightly privileged. The next series of experiments was carried on for σ described by the following distribution:

$$\sigma(S) = L \times (\eta(l(S)/L) + U(0,2)) \quad (6.11)$$

which in turn slightly favoured the stations located close to the ends of the bus. These observations suggested to look for σ among distributions of the following type:

$$\sigma(S) = L \times (\tau \times \eta(l(S)/L) + U(0, 2 \times (1-\tau) \times \eta(l(S)/L) + \alpha)) \quad (6.12)$$

where $\tau \in [0,1]$ and $\alpha \geq 0$ are distribution parameters. No backoff method was used: a colliding (losing) station was allowed to retransmit immediately after sensing the bus idle and obeying the inter-packet spacing rules. The most spectacular behaviour of the reference network was observed for $\alpha = 1$ and $\tau = 0.8$ (fig. 10). Similar results were obtained for other versions of *DCS*, i.e., other cable lengths and numbers of stations.

7. Eliminating the Unfairness of SCS

The *SCS* network combines certain properties of *DCS* and Ethernet. Assume that a station *A* has a ready packet to be transmitted, and is waiting for the beginning of a silence period. Without loss of generality, we may assume that *A* wants to transmit to the left. Let *B* be another station which is also waiting for silence to start its transfer attempt. If *B* transmits to the left then the scenario is exactly as in *DCS*, otherwise we have essentially an Ethernet-type collision. It may happen that neither *A* nor *B* loses in this collision; however, if one station is preempted by the other, it happens according to the same rules as in Ethernet.

7.1. The Unfairness of Ethernet

Thus let us consider pure Ethernet first. It turns out that Ethernet is also an unfair network (see fig. 11), although its unfairness is not so serious as for the segmented networks. As a byproduct of solving the problem of poor fairness of *SCS* we will show how to solve the corresponding problem for Ethernet.

In Ethernet, a station A located in the middle of the cable is privileged over a station B situated at one end of the bus, because the cable seems shorter to A than to B. The maximum distance from A to any other station is $L/2$, whereas for B that distance is L. If we assume that stations are uniformly distributed along the bus, then the average distance from A to another station is $L/4$, and the average distance from B to another station is $L/2$. Thus, on the average, station A learns faster about any changes in the status of the channel than B. A takes advantage of that fact by starting its activities sooner and thus preempting stations that are less fortunately located.

Let us assume that two stations A and B receive packets to be transmitted at (approximately) the same time. If the packets arrive when the bus has been idle for the duration of time corresponding to the inter-packet space, then the competition between A and B is fair. The unfairness of Ethernet manifests itself under heavy traffic conditions. Thus let us assume that the bus is not idle and A and B are forced to wait: first until they detect silence in the channel, and next until the inter-packet spacing rules are obeyed. Due to the fact that A and B are connected to different places of the bus, silence in the channel is perceived by them at different times (usually). Let us assume that A learns sooner than B about the beginning of the silence period. If A and B use inter-packet spaces of approximately the same length then A is privileged. The reason is that A starts its transfer sooner and now it can either preempt B (if B detects the transfer of A before deciding to transmit its own packet)[5], or become involved in a collision with B (if B starts its transfer before detecting the activity of A). Thus B can at most collide with A whereas A has a chance for transmitting its packet successfully. This phenomenon manifests itself clearly in figure 11. Stations located closer to the middle of the bus learn (on the average) sooner about silence periods in the channel and are privileged over stations located at the ends of the cable. In the scenario described above, A should use a longer inter-packet space than B − so that the chances of A and B are equal.

Let us assume the notation from the previous section and suppose for simplicity that the silence perceived by A and B has originated from one well-defined place on the bus. We can view this place as a virtual station C that terminates some activity. In the case when the silence period has been caused by the end of a successful transfer, C is a real station and the point of origin of the silence period is actually well defined. Let t_s be the time moment when the virtual station C terminates its activity. If stations A and B are ready to send packets, they will start their transfer attempts at:

$$t_A = t_s + d(A, C) + \Pi(A) \qquad \text{and} \tag{7.1.1}$$

$$t_B = t_s + d(B, C) + \Pi(B)$$

respectively. We would say that their chances for a successful transfer are equal if:

$$t_A = t_B \tag{7.1.2}$$

As we have already mentioned in section 6, the above requirement cannot be fulfilled absolutely, as it requires that A and B know their mutual locations and the location of C. However, similarly as in section 9, we can consider **weak fairness**: postulate that equality 7.1.2 hold "on the average", i.e., for each station A

$$E(t_A - t_B) = 0 \tag{7.1.3}$$

where the expectancy E is taken over all potential Bs and Cs. The difference between t_A

and t_B depends on neither t_s nor π. Therefore, the above equation can be rewritten as:

$$E(\sigma(A) - \sigma(B) + d(A, C) - d(B, C)) = 0 \tag{7.1.4}$$

Unfortunately, unlike in *DCS*, even weak fairness of Ethernet is difficult to enforce. Namely, it is clear that the value of $\sigma(S)$, for a station S, must depend not only on the location of S, but also on the traffic pattern. In reality, after a collision, different stations may hear silence periods originated from different points on the cable, and the assumption that each silence has a well-defined origin is not valid.

We can assume that whenever S is to obey the inter-packet space after sensing a period of silence, it first determines whether the silence period follows a collision or a successful transfer. Then it generates the length of the space, denoted by $\sigma_c(S)$ and $\sigma_t(S)$, respectively, according to some distributions with the mean values of $\eta_c(S) = E(\sigma_c(S))$ and $\eta_t(S) = E(\sigma_t(S))$. We would like to determine how the values of $\eta_c(S)$ and $\eta_t(S)$ should depend on the location of S for the idealistic network model consisting of infinitely many stations densely and uniformly distributed along the [0,1] interval.

Let us look at $\eta_t(S)$ first. Under the assumption that the traffic is uniform, the postulate expressed in equation 7.1.4 boils down to the requirement that for each $S \in [0,1]$

$$\int_0^1 \int_0^1 \eta_t(S) - \eta_t(B) + \left|S - C\right| - \left|B - C\right| \, dC \, dB = 0 \tag{7.1.5}$$

It can be easily determined that the above equation[6] is satisfied by any function of the following form:

$$\eta_t(S) = S \times (1 - S) + \delta \tag{7.1.6}$$

where δ is an arbitrary constant. Thus we reach the final formula:

$$\eta_t(S) = S \times (1 - S) \tag{7.1.7}$$

Note that equation 7.1.7 is in agreement with intuition: η_t is symmetric and its values grow as S approaches the middle of the cable.

The values of η_c are more difficult to determine. Instead of calculating η_c directly, we will determine the proper value of η_c^k, assuming that the collision involves exactly k stations. Let us assume that the k stations $S_1,.., S_k$ involved in a collision started their transfer attempts at the same time. In general it need not be true, however, this is exactly the situation that occurs "on the average", if the k colliding transfer attempts follow a successful transmission (equations 7.1.5 and 7.1.6). Under this assumption, the silence period following the collision originates from one point:

$$C = \frac{\max_k(S_1,.., S_k) + \min_k(S_1,.., S_k)}{2} \tag{7.1.8}$$

If the traffic is uniform, $S_i,\ i=1,..,k$ can be viewed as k independent uniformly distributed random variables. Let $h(S) = S$, if $S \le 1/2$ and $1 - S$, otherwise. The density function of the distribution of C is then given by:

$$\xi_k(C) = k \times (2h(C))^{k-1} \tag{7.1.9}$$

If we assume that a station S detecting a silence period after a collision knows how many parties have been involved in it, the condition for $\eta_c^k(S)$ can be expressed as:

$$\int_0^1 \int_0^1 \xi_k(C) \times \left\{ \eta_c^k(S) - \eta_c^k(B) + \left| S - C \right| - \left| B - C \right| \right\} dC\, dB = 0 \tag{7.1.10}$$

After simple calculations we determine that equation 7.1.10 is satisfied by the following function:

$$\eta_c^k(S) = h(S) - \frac{(2h(S))^{k+1}}{k(k+1)} + \delta \tag{7.1.11}$$

where δ is a constant.

Of course, in general, a station sensing a collision is not able to tell how many parties have been involved in it. Let us note, however, that the simple function:

$$\eta_c(S) = h(S) \tag{7.1.12}$$

is a good k-independent approximation of η_c^k with minimum δ, especially for higher values of k − when the traffic is heavier. Even for $k = 2$, the remaining part of equation 7.1.11 is a third order function of S.

The way $\eta = E(\sigma)$ was determined says nothing about the adequate distribution of σ in a real-life network (of length L). Simulation experiments conducted for the uniform traffic pattern indicate that Ethernet operates quite nice for unrandomised σ, i.e.:

$$\sigma(S) = \eta(l(S)/L) \times L \tag{7.1.13}$$

(see figure 12)[7]. The role of the "randomising agent" is played here by the Binary Exponential Backoff Algorithm. The global average performance of Ethernet with the location-specific packet spacing adjustment was not observably worse than the performance of the commercial network.

7.2. The Unfairness of SCS

Let us now return to the *SCS* network. In the light of the observations made at the beginning of section 7 and in section 7.1, the mean length of inter-packet space for a station A should depend on the direction of intended transfer and should take into account whether the silence period follows a successful transfer or a collision. Two problems appear here.

The first of them is that (according to the protocol outlined in section 2, point SCS-1) a transmitting station jams the unused part of the channel. Thus, for a station located in that part, it is difficult to tell a valid transfer from a collision. One possible solution would be to monitor the length of jamming signals. A collision jam is much shorter than the jam inserted during the transmission of a packet. Another solution (suggested in section 2) is to replace the jamming signal in question with a copy of the packet being transmitted. The sole purpose of the jam is to inform the stations located in the other part of the channel that a transfer is in progress, and any activity inserted into the unused part of the bus will do as well.

The other problem is that in *SCS* two valid transfers can be active at a time, i.e., two stations *A* and *B*, *A* located on the left of *B*, *A* transmitting to the left and *B* transmitting to the right. Thus the silence period following the end of a successful transfer need not originate in one place. A station detecting the end of a successful transfer is generally not able to tell whether there was another simultaneous transfer in the opposite direction. Let us consider the idealistic *SCS* network consisting of infinitely many stations densely and uniformly populated along the bus and assume that the traffic is uniform. The probability that two randomly chosen stations want to transmit their packets according to the above scenario is 1/9. Thus we will simply ignore it[8].

Assuming that the silence period is a consequence of a single successful transfer, we get the following equation for η_t:

$$\int_0^1 \left\{ \int_0^1 \eta_t(S) - \eta_t(B) + \left| S - C \right| - \left| B - C \right| + B(S - B)\, dB \right\} dC = 0 \quad (7.2.1)$$

The probability that station *B* wants to transmit to the left is equal *B*. Thus the term $B(S - B)$ represents the case when *B* transmits to the left (see equations 6.5 and 7.1.5). Similarly, the equation for η_c^k takes the following form:

$$\int_0^1 \xi_k(C) \times \left\{ \int_0^1 \eta_c^k(S) - \eta_c^k(B) + \left| S - C \right| - \left| B - C \right| + B(S - B)\, dB \right\} dC = 0 \quad (7.2.2)$$

where $\xi_k(C)$ is given by equation 7.1.9. Equation 7.2.1 is satisfied by

$$\eta_t(S) = S(1/2 - S) + \delta \quad (7.2.3)$$

where the minimum δ for which the function is nonnegative is 1/2. Let us note that the shape of $\eta_t(S)$ agrees with intuition. Namely, the minimum value is reached for $S = 1$, and as S moves toward the left end of the cable, the function grows and reaches its maximum for $S = 1/4$. The value at $S = 0$ is 1/2 which is less than for $S = 1$. Clearly, a station located at the right end of the cable and transmitting a packet to the left should enjoy the highest privilege – its location-dependent inter-packet space should be the shortest. For a station *S* situated close to the left end of the cable and transmitting to the left we have a trade-off between its low privilege – with respect to other stations transmitting to the left, and a high privilege – with respect to the stations transmitting to the right (which can preempt the transmission of *S*). This explains why $\eta_t(S)$ is non-symmetric and why its values at the two ends of the cable are different. Similarly, equation 7.2.2 is solved by

$$\eta_c^k(S) = h(S)\times\frac{k}{k+1} - \frac{(2h(S))^{k+1}}{2(k+1)} - \frac{S}{2} + \delta \qquad (7.2.4)$$

To obtain a *k*-independent approximation of η_c^k we note that $k/(k+1)$ converges to 1 and the last term of equation 7.2.4 converges to 0 as k grows. Thus we have

$$\eta_c(S) = h(S) - \frac{S}{2} + \delta \qquad (7.2.5)$$

The minimum δ making η_c non-negative is $1/2$. Let us note that the shape of such η_c is similar to the shape of η_t (equation 7.2.3).

The experiments conducted for the *SCS* network with the packet spacing adjustment described by η_t and η_c, have shown a significant improvement in the observed fairness of the network. The results presented in figure 13 were obtained for the uniform traffic and unrandomised σ (equation 4.1.14). The backoff function was the same as in commercial Ethernet.

8. Summary

We have presented two networks based on the Ethernet concept with slightly modified notion of the *ether* - the common medium accessible to all stations. In our proposal, the *ether* is not continuous all the time. Although most of its properties are retained (essentially it behaves like a uniform commonly accessible and connected medium) the new *ether* may be divided into separate (non-interfering) segments. This idea, which seems quite natural when the *ether* is implemented as a (coaxial) cable, substantially improves the performance of the collision protocol. The nature of the proposed modification is fairly simple. Therefore, one could believe that there will be no serious implementation problems.

One of the two proposed networks (*DCS*) is especially interesting. Its maximum asymptotic throughput turns out to be equal to the capacity of the medium. In some sense *DCS* gets us exactly as far as we can ever get with a bus-type network.

We have also discussed the influence of the station location on its relative priority in the proposed networks and suggested some methods of improving their fairness. As a byproduct of this demonstration, we have shown how to equalise stations' priorities in commercial Ethernet.

The properties of the new networks have been analysed by simulation. It definitely deserves to be emphasised once again that the model used for simulation was accurate and reflected all the relevant aspects of a real network. Thus it is felt that the results presented in this paper can be transformed directly into reality.

REFERENCES

[1] Chlamtac, I., Franta, W.R., Levin, D.: BRAM: The Broadcast Recognizing Access Method. IEEE Transactions on Communication, vol. COM-27, pp. 1183-1190, Aug. 1979.

[2] Garret, M.W., Li, S.Q.: A Study of Slot Reuse in Dual Bus Multiple Access Networks. IEEE Journal on Selected Areas in Communications, vol. SAC-9, no. 2, pp. 248-256, 1991.

[3] Gburzynski, P., Majithia, J.C., Wilson, T.C.: An Improved Backoff Algorithm for Ethernet-type Local Area Networks. Technical Report TR87-9, Department of Computing Science, University of Alberta, Edmonton, Canada, 1987.

[4] Gburzynski, P. and Rudnicki, P.: LANSF: A Protocol Modelling Environment and its Implementation, Software Practice and Experience, vol. 21, no. 1, January 1991, pp. 51-76.

[5] Gburzynski, P. and Rudnicki, P.: Bounded Packet Delay Protocols for CSMA/CD bus: Modeling in LANSF, Proceedings of the 19'th Annual Pittsburgh Conference on Modeling and Simulation, May 1988, pp. 1185-1194.

[6] Gburzynski, P. and Rudnicki, P.: Modeling Low-level Communication Protocols: A Modular Approach, Proceedings of the 4'th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation, Palma de Mallorca, September 1988, pp. 89-108.

[7] Gburzynski, P. and Rudnicki, P.: A Better-than-Token Protocol with Bounded Packet Delay Time for Ethernet-type LANs, Proceedings of the 1987 Symposium on the Simulation of Computer Networks, Colorado Springs, August 1987, pp. 110-117.

[8] Goodman, J. *et al.*: Stability of Binary Exponential Backoff, Journal of the ACM, vol. 35, no. 3, July 1988, pp. 579-602.

[9] Gopal, P.M., Wong, J.W.: Analysis of a Hybrid Token-CSMA/CD Protocols for Bus Networks. Computer Networks and ISDN Systems 9, pp. 131-141, 1985.

[10] IEEE Project 802.6, Proposed Standard, Distributed Queue Dual Bus Metropolitan Area Network. Doc. No. P802.6/D15, October 1990.

[11] Jain, R.: Performance Analysis of FDDI Token Ring Networks: Effect of Parameters and Guidelines for Setting TTRT. Technical Report DEC-TR-655, Digital Equipment Corporation, September 1989.

[12] Kiesel, W.M., Kuehn, P.J.: A New CSMA/CD Protocol for Local Area Network with Dynamic Priorities and Low Collision Probability. IEEE Journal on Selected Areas in

Communications, vol. SAC-1, no. 5, pp. 869-876, 1983.

[13] Kleinrock, L.: On the Behavior of a Very Fast Bidirectional Bus Network. Proceedings of the 1987 IEEE International Communications Conference, June 1987, Seattle WA., pp. 1419-1426.

[14] Maxemchuk, N.F.: Twelve Random Access Strategies for Fiber Optic Networks. IEEE Transactions on Communications, vol. 36, no. 8, pp. 942-950, 1988.

[15] S.K. Mehra, Design and Evaluation of a Class of Reconfigurable Computers, Ph.D. Thesis, University of Waterloo, Waterloo, Ont., 1980.

[16] Metcalfe, R.M. and Boggs, D.R.: Ethernet: Distributed Packed Switching for Local Computer Networks, CACM 19, no. 7, pp. 395-403.

[17] Molloy, M.K.: Collision Resolution on the CSMA/CD Bus. Computer Networks and ISDN Systems 9, pp. 209-214, 1985.

[18] Moura, J.A.B.: Loops and Ethernet: Evaluation and Comparison of Performance and Complexity. Computer Communications Network Group T-Report, T-80, University of Waterloo, Waterloo, Ont., 1979.

[19] Moura, J.A.B., Field, J.A., Wong, J.W.: Simulation of Collision Control Algorithms in Ethernets, Sixth Data Communications Symposium. Asilomar Conference Grounds, Pacific Grove, Ca., 1979.

[20] Rodrigues, M.A.: Erasure Node: Performance Improvements for the IEEE 802.6 MAN. Proceedings of INFOCOM'90, June 1990, pp. 636-643.

[21] Shoch, J.F. *et al.*: Evolution of the Ethernet Local Computer Network, Computer, Aug. 1982.

[22] Stallings, W. (ed.): Local Network Technology. IEEE Computer Society Press, 1984.

[23] Stallings, W.: A Tutorial on the IEEE 802 Local Network Standard. In: Pickholtz, R.L. (ed.): Local Area & Multiple Access Networks. Computer Science Press, pp. 1-30, 1986.

[24] Tanenbaum, A.S.: Computer Networks. Prentice-Hall, 1981, pp. 296-306.