Performance analysis of FDDI LANs using numerical methods

B. Albert
A.P. Jayasumana

Indexing terms: Fibre distributed data networks, FDDI follow-on, Token ring networks

Abstract: Simulators are valuable tools; they help in understanding how complex systems will operate and perform without actually having to build a physical model to test. However, they often require extensive development and run times. This is especially true for many networks such as FDDI, where a large number of nodes and priority classes are involved. A high-level algorithm is presented to evaluate the performance characteristics of a token LAN. The goal of the research was to find a method for obtaining an estimate of the average access delay, and the average end-to-end delay of advanced token-ring networks. The FDDI (fibre distributed data interface) standards were used as the basis of a FFOL (FDDI follow-on LAN) solution. A method was developed that will generate exact results for such networks at low- to medium-load network environments. As the load level increases, the error rate of the estimate also increases; this however is monitored and can be used as a confidence measure for the results obtained. A symmetric multimedia environment is used in the model, with bimodal traffic consisting of short data packets, and longer video packets. A short introduction to the network setup and the examination of the method developed is presented in single and multinetwork environments. Specific network parameters were chosen for these experiments, but the method is easy to customise to other network parameters.

1 Introduction

Network capacity demand continue to increase and there are two directions to expand in: increase the operating speed, or interconnect multiple networks. The choice is not obvious, and involves trade-offs and limitations on the range of scalability and its applicability to different traffic patterns. There are many design approaches, each has its advantages and drawbacks. Many of these trade-offs can be successfully explored with simulations.

This paper presents a technique to obtain performance estimates by finding a discrete solution that meets the input criteria. The technique described is not as computation intensive nor difficult to implement in comparison to a discrete event simulator for evaluating the performance. The advantage of this method is that it presents fast results. The degree of accuracy can be controlled by the number of states implemented in the simulations.

Other computational approaches to network performance evaluation have been presented, some dealing with token ring network performance, mostly using statistical methods [1–3].

The simulator itself is designed to be flexible; it can be easily changed to accommodate different experiments. It is written in a high-level language, Mathematical [4], to facilitate easy verification and easy output plotting. All the customisation is done at the beginning of the program in the section containing the operating parameters and constants. An important consideration in a multinetwork environment is data locality; this also is programmable to simulate different network loading conditions [5].

![FDDI network structure](image)

**Fig.1  FDDI network structure**

2 Background on FDDI and FFOL

The FDDI and FFOL standards were developed in the American National Standards Institute (ANSI) Com-
committee X3T12 (formerly X3T9.5). The FDDI network is defined as having a dual ring-of-trees topology, shown structurally in Fig. 1. There are the primary and secondary rings which together form the dual-ring part of the FDDI network [6]. These rings are shown as the rectangles at the top of the Figure. The tree structure unfolds below the main dual ring. To be a network it must contain at least one node with a MAC (media access controller) in it and no more than a total of 1,000 MACs. There should be a dual ring although the network will function with only a single ring. The standard defines the minimum network delay as 40 bits x 8 bit/ns = 320 ns, which is equivalent to the delay caused by 63m of optical fibre. The maximum network length is 100 km for each of the primary and secondary rings, which result in a total network length of 200 km in the case of a wrapped network. The standard does not specify any limitations on the distribution of MACs between the primary and secondary rings.

There are two basic building blocks for the FDDI network, stations and concentrators. Stations are the end-user nodes. Concentrators are network management nodes that do not directly affect the end user. Concentrators are used in the network structure to branch out the ring below the dual ring. In Fig. 1 the stations and concentrators are represented by boxes labelled S and C, respectively.

The FDDI standards define five basic types of nodes. These nodes can be classified into two groups. One group is the station-type node group, which is the actual user attachment point to the network. The other group is the concentrator-type node group that can be used to attach stations and other concentrators to the network in a dynamic manner. Both of these node types can be either single attachment or dual attachment. The five node types are:

- dual attachment station
- single attachment station
- dual attachment concentrator
- single attachment concentrator
- null attachment concentrator

The main ring of an FDDI network allows only dual-attachment nodes to be directly connected to it. Dual-attachment nodes make optical connection to both the primary and secondary optical paths of the main ring directly. To keep the main ring operational at all times, dual-attachment nodes should have optical bypass relays to isolate a faulty node. Single-attachment nodes connect to only one of the two optical rings. They are connected to the main ring through concentrators and thus do not directly affect the availability of the main ring and they need not be equipped with optical bypass switches. This lowers the cost of the single-attachment node, although this saving is offset by the need to have a concentrator.

FDDI-II is a further development of FDDI that includes isochronous traffic. It requires new hardware but is capable of carrying FDDI compatible traffic in a background mode. The two types of FDDI networks can be interconnected with the obvious limitation that FDDI cannot carry FDDI-II isochronous traffic.

3 Method for estimating access delay in token ring network environment at different load levels

In this Section an explanation of the method used to obtain performance results is presented, some confidence checks are introduced, and the importance of the major operating parameters are given. This example is for a single network only, a 100 bit/s FDDI token ring network, which uses a bit-clock of 10 ns. The technique described has been developed over time [7–10].

It was observed in previous experiments with token ring networks at low-load levels that most of the time the network was idle. What happens is that the token is rotating idly and a packet comes along and wants access to the network. This packet has immediate access to the network (in half of an idle-token rotation time, on average), it then gets transmitted. At this point it is likely that the network will return to the idle state. Sometimes there are two packets that get transmitted in a token rotation, but the probability of larger number packets in a given rotation continuously decreases, and quickly reaches zero (at low-load levels).

The idle-token rotation time is the sum of the cable delay, device delay, and the token time for a FDDI network with 20 km of cable and 100 devices,

\[
\text{CableTime} = 20 \text{ km} \times 5.085 \frac{\mu s}{\text{km}} = 102 \mu s
\]

\[
\text{NodeDelayTime} = 100 \text{ devices} \times 0.690 \frac{\mu s}{\text{km}} = 69 \mu s
\]

\[
\text{TokenTime} = 88 \text{ bits} \times 10 \frac{\text{ns}}{\text{bits}} = 1 \mu s
\]

\[
\text{TokenRotationTime}_{\text{idle}} = \text{CableTime} + \text{NodeDelayTime} + \text{TokenTime} = 163 \mu s
\]

The inverse of the idle-token rotation time is the number of token rotations per second. In the idle network of our example the number of token rotations would be

\[
\text{NumTokenRotations} = \frac{1}{\text{TokenRotationTime}_{\text{idle}}} = 6135 \text{ rotations}
\]

We simulate a mixture of network traffic consisting of video streams and data packets; this problem has been addressed by other authors in a similar manner [2, 11–14]. The amount of video and data traffic generated is controlled by the load factor (LF) and at LF = 1 there is 15 Mbit/s of video traffic and 15 Mbit/s of data traffic, whereby this does not include the packet overhead. Table 1 gives a summary of the data and video packet parameters at LF = 1. The results at different load factors are obtained by stepping the simulator through the range of LF = 0 to LF = 2 in steps of 0.1 (21 total steps). We limit the simulations to a maximum LF of two because this creates a reasonable load for the method described; a higher load level can be simulated at the expense of adding more states to the system.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data</th>
<th>Video</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data per packet</td>
<td>1000 bits</td>
<td>10000 bits</td>
</tr>
<tr>
<td>Overhead per packet</td>
<td>224 bits</td>
<td>224 bits</td>
</tr>
<tr>
<td>Total packet size</td>
<td>1224 bits</td>
<td>10224 bits</td>
</tr>
<tr>
<td>Packet time</td>
<td>12.5 μs</td>
<td>102.5 μs</td>
</tr>
<tr>
<td>Packets per second</td>
<td>15000</td>
<td>1500</td>
</tr>
<tr>
<td>Average interarrival time</td>
<td>67.5 μs</td>
<td>667.5 μs</td>
</tr>
<tr>
<td>Data rate</td>
<td>15 Mbit/s</td>
<td>15 Mbit/s</td>
</tr>
<tr>
<td>Overhead rate</td>
<td>3.36 Mbit/s</td>
<td>0.34 Mbit/s</td>
</tr>
<tr>
<td>Offered load</td>
<td>18.36 Mbit/s</td>
<td>15.34 Mbit/s</td>
</tr>
</tbody>
</table>

Assuming a linear, random distribution of packets across time and devices, the mean arrival rate for data packets at one every 67μs and for video packets at one per 667μs. This leads to an interesting and important observation: on average, the number data packet per idle-token rotation is given by

\[
\text{DataPackets} = \frac{\text{TokenRotationTime}_{idle}}{\text{DataInterarrivalTime}} = \frac{163 \ \mu s}{67 \ \mu s} = 2.4
\]

These packets are short (12μs) and are simply included in the updated idle-token rotation time; they do not otherwise affect the operation of the network. Adding additional time to the idle-token rotation time with the background data packets increases the amount of data packets that will be sent in an infinite spiral. This is accounted for in the program with a special expansion factor that is part of MFAC (expansion multiplication factor). Video packets are less frequent, and assuming only a single video packet would occur, the average number of video packets per idle token rotation can be calculated

\[
\text{VideoPackets} = \frac{\text{TokenRotationTime}_{idle}}{\text{VideoInterarrivalTime}} = \frac{163 \ \mu s}{667 \ \mu s} = 0.24
\]

There is only a 24% probability that a video packet arrives during an idle token rotation. If a video packet is transmitted the token rotation time is lengthened and the probability of an additional video packet in the next token rotation is increased.

The video packets are long: their transmission time is almost the same as the idle token rotation time. It is also possible that more than one video packet is available for transmission on a given token rotation. A state machine can be defined with the number of video packets transmitted as the state number and the interstate probabilities given by \( P_{xy} \), shown in Fig. 2.

![Fig. 2 Video packet transmission state machine](image)

The Figure shows only the state transitions that affect state 0 (\( S_0 \)), there are six times as many state transitions corresponding to each of the states \( S_0 \) to \( S_5 \). The transition probability \( P_{01} \) is used to refer to the probability of a transition to state 1 given that the state machine is currently in state 0. The compute time required per simulation run increases significantly with the number of states and must be traded off against accuracy, additional states to not significantly extend the useful range of the simulator. Six states were used for this simulation.

For each state the transition probabilities to the next state are binomially distributed as a function of the load factor and the current state. These are calculated in the program in the section with the comment starting ‘(*F*)’ (this is the general format used for cross referencing to the program sections). A full program listing is included in [6, 7]. The following equations apply to state 0 (no video packets):

\[
\text{prob} 0 = \text{VideoPacketLF1} \times \text{LoadFactor} \\
\times (\text{TokenRotation}_{idle} + \text{DataDelay} + \text{VideoDelay})
\]

\( P_{00} \) = binomial distribution \( (\text{prob} 0, 0) \)

\( P_{01} \) = binomial distribution \( (\text{prob} 0, 1) \)

\( P_{02} \) = binomial distribution \( (\text{prob} 0, 2) \)

\( P_{03} \) = binomial distribution \( (\text{prob} 0, 3) \)

\( P_{04} \) = binomial distribution \( (\text{prob} 0, 4) \)

\( P_{05} \) = binomial distribution \( (\text{prob} 0, 5) \)

where VideoPacketsLF1 is the number of video offered to the system per second at load factor = 1, and prob0 is the average probability of having a video packet in state 0 (0.24) in this case. \( P_{00} \) through \( P_{05} \) are the specific state transition probabilities calculated using a binomial distribution with parameters \( (\text{prob} x, y) \) where \( \text{prob} x \) is the average probability of having a video packet in state \( x \) and \( y \) is the number of video packets to be transmitted in this state (this then becomes the next state of the state machine \( \text{St Treat} \)).

This procedure is repeated for all six states and the following set of linear equations can be defined:

\[
S_0 \times P_{01} + S_1 \times P_{10} + S_2 \times P_{20} + S_3 \times P_{30} + S_4 \times P_{40} + S_5 \times P_{50} \\
= S_0 \times (P_{00} + P_{01} + P_{02} + P_{03} + P_{04} + P_{05})
\]

\[
S_0 \times P_{03} + S_1 \times P_{11} + S_2 \times P_{21} + S_3 \times P_{31} + S_4 \times P_{41} + S_5 \times P_{51} \\
= S_1 \times (P_{10} + P_{11} + P_{12} + P_{13} + P_{14} + P_{15})
\]

\[
S_0 \times P_{02} + S_1 \times P_{12} + S_2 \times P_{22} + S_3 \times P_{32} + S_4 \times P_{42} + S_5 \times P_{52} \\
= S_2 \times (P_{20} + P_{21} + P_{22} + P_{23} + P_{24} + P_{25})
\]

\[
S_0 \times P_{03} + S_1 \times P_{13} + S_2 \times P_{23} + S_3 \times P_{33} + S_4 \times P_{43} + S_5 \times P_{53} \\
= S_3 \times (P_{30} + P_{31} + P_{32} + P_{33} + P_{34} + P_{35})
\]

\[
S_0 \times P_{04} + S_1 \times P_{14} + S_2 \times P_{24} + S_3 \times P_{34} + S_4 \times P_{44} + S_5 \times P_{54} \\
= S_4 \times (P_{40} + P_{41} + P_{42} + P_{43} + P_{44} + P_{45})
\]

\[
S_0 \times P_{05} + S_1 \times P_{15} + S_2 \times P_{25} + S_3 \times P_{35} + S_4 \times P_{45} + S_5 \times P_{55} \\
= S_5 \times (P_{50} + P_{51} + P_{52} + P_{53} + P_{54} + P_{55})
\]

This is a complete set of linear equations and could easily be solved using \( \sum_{x=0}^{5} P_{0x} = 1 \). This would guarantee delivery of all data packets at the expense of video packet loss. The reason for this is that the model as presented truncates the maximum number of video packets transmitted during one token rotation to six, whereas the probability generator mechanism does not, and many of the video packets are lost at high load levels. The model, however, requires that all video packets be delivered at the expense of dropping data packets.
This can be achieved by substituting the following equation for the last of the equations and then solving for the states \((St(x))\):

\[
0 + St \times 1 + St_2 \times 2 + St_3 \times 3 + St_4 \times 4 + St_5 \times 5
\]

\[
= VideoPacketsLF1 \times LoadFactor
\]

The effect of this is that all the occurrences of states beyond state 5 are lumped together in state 5. This results in all video packets being delivered at the expense of data packets, as the FDDI protocol requires. This is calculated in the program in the section with the comment starting \("G\). A histogram for the frequency of states in this example is given in Fig. 3, at 60% offered load. By comparison, Fig. 4 shows the state distribution at an offered load of 40%. Notice the dominance of idle token (0 video packets).

![Fig. 3 State frequency histogram at 60% offered load](image)

![Fig. 4 State frequency histogram at 60% offered load](image)

The average access delay can be computed:

\[
AveAccessDelay = \sum_{x=0}^{5} \frac{(NumPackets(x) \times St(x) \times StateDel(x))}{2}
\]

This is calculated in the program in the section with the comment starting \("H\), and the other data points are logged in \("I\) and plotted in \("K\). The average access delay is plotted in Fig. 5 and the average end-to-end delay is calculated as:

\[
AveEndToEndDelay = AveAccessDelay
\]

\[\frac{\sum_{x=0}^{5} AvePacketLength}{TokenRotation_{idle}} + \frac{MFac}{0.9}
\]

The average end-to-end delay is plotted in Fig. 6. The total network load includes overhead and is larger than the offered load, and is plotted in Fig. 7.

![Fig. 5 Average access delay against offered load](image)

![Fig. 6 Average end-to-end delay against offered load](image)

There are two checks that are performed on the resulting data. First the percentage of data delivered can be calculated,

\[
DataDelivered = \frac{\sum_{x=0}^{5} St(x) \times DataPackets(x)}{DataPacketLF1 \times LoadFactor}
\]

This shows the overall error rate of the estimation and is plotted in Fig. 7 and in the program in the section with the comment starting \("L\). The portion of video

delivered, which should always be 100% (by design), is calculated:

\[
Video_{\text{Delivered}} = \frac{\sum_{x=1}^{5} St(x) \times (x)}{VideoPacketLF \times LoadFactor}
\]

This is plotted in Fig. 7 and is calculated in the program in the section with the comment starting ‘*L2’.
Finally, the network utilisation is calculated in ‘*M’ (see Fig. 7).

![Fig. 7 Network utilisation](image)

![Fig. 8 Data delivery rate](image)

To verify the results obtained by the simulator a comparison study was done with another method published by Werahera [15-17]. This author himself had been involved in the comparison study and was a well-known expert [18]. The comparison method is an analytical solution based on an estimate for the average waiting time for the token

\[
AveWaitTime = \frac{pX^2}{2X(1-p)} + C_0 \left( \frac{1 - \frac{p}{N_{\text{num}}} - \frac{C_0}{p}}{2 \left( 1 - \frac{p}{N_{\text{num}}} \right)} \right)
\]

The operating parameters are summarised in Table 2. The results are shown in Fig. 7 and the results are very encouraging. The differences are due to the comparison model only using one data type whereas our simulations are bimodal. The video packets are present at all load levels and affect the access time owing to their length.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>p</td>
<td>Offered load</td>
<td>0-0.6</td>
</tr>
<tr>
<td>X</td>
<td>Mean message size</td>
<td>2044 bits</td>
</tr>
<tr>
<td>C_0</td>
<td>Idle TRT</td>
<td>163μs</td>
</tr>
<tr>
<td>Num</td>
<td>Number of stations</td>
<td>100</td>
</tr>
<tr>
<td>T</td>
<td>Target TRT</td>
<td>1ms</td>
</tr>
</tbody>
</table>

To verify the results obtained by the simulator a comparison study was done with another method published by Werahera [15-17]. This author himself had done an extensive verification and comparison to a well-known method [18]. The comparison method is an analytical solution based on an estimate for the average waiting time for a token

\[
AveWaitTime = \frac{pX^2}{2X(1-p)} + C_0 \left( \frac{1 - \frac{p}{N_{\text{num}}} - \frac{C_0}{p}}{2 \left( 1 - \frac{p}{N_{\text{num}}} \right)} \right)
\]

The operating parameters are summarised in Table 2. The results are shown in Fig. 7 and the results are very encouraging. The differences are due to the comparison model only using one data type whereas our simulations are bimodal. The video packets are present at all load levels and affect the access time owing to their length.

Table 2: Operating parameters for analytical comparison

4 Performance of hierarchically interconnected networks

The algorithm presented can be extended to symmetric multinetworx. Hierarchical FDDI is a proposed network interconnection scheme which allows the continued use of existing FDDI and FDDI-II networks connected into a single network structure. This scheme was proposed to the ANSI standards committee for consideration as a FFOL proposal.

In an H-FDDI environment, a multinetworx configuration exists where each network can access each other network directly. This can be viewed as having an extra node on the network that acts as an interface to the rest of the networks—a bridge to the rest of the network, this is shown in Fig. 11. The portion of data traffic that leaves the local network through the bridge is considered nonlocal and by symmetry an equal number of data packets will be destined for the local network from the rest of the network. This is referred to as the nonlocality factor and can be used to measure the performance of a multinetworx. The simulator is capable of giving comparative results for three different locality factors per simulation run. In the following
example eight networks are interconnected. The operating parameters of this multinet work configuration are given in Table 3, and the average access delay of this networking systems presented in Fig. 12. The three locality factors used are 90, 70, and 50% representing networks with decreasingly local traffic, resulting in more of the traffic that does not have a destination on the source network.

Table 3: Multinet operating parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>H-FDDI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of networks</td>
<td>8</td>
</tr>
<tr>
<td>Data rate per network</td>
<td>100Mbps/s</td>
</tr>
<tr>
<td>Total data rate</td>
<td>800Mbps/s</td>
</tr>
<tr>
<td>Number of user nodes</td>
<td>100</td>
</tr>
<tr>
<td>Number of system nodes</td>
<td>8</td>
</tr>
<tr>
<td>Total number of nodes</td>
<td>108</td>
</tr>
<tr>
<td>Delay per node</td>
<td>600ns</td>
</tr>
<tr>
<td>Total user fibre length</td>
<td>20km</td>
</tr>
<tr>
<td>System fibre length</td>
<td>8km</td>
</tr>
<tr>
<td>Total fibre length</td>
<td>28km</td>
</tr>
</tbody>
</table>

Fig. 12 Average access delay for multinet environment

5 Summary

The proposed simulator model takes advantage of the discrete nature of the behaviour that token ring networks exhibit at low-load level. The average access delay for data and video packets is plotted against offered load and exact results can often be obtained for up to 50% of the available bandwidth. Two service classes are modelled, corresponding to the FDDI synchronous and asynchronous classes that behave in the intended manner; in case of overload, the asynchronous class is degraded. The simulator has built-in flexibility to model advanced token ring networks including H-FDDI and other FFOL network designs.

6 References


154