A Modeling and Simulation Method for Transient Traffic LAN

PAPER

A Modeling and Simulation Method for Transient Traffic LAN

Susumu ISHIHARA[†], *Student Member and* **Minoru OKADA**[††], *Member*

SUMMARY

In this paper a protocol-based modeling and simulation method of performance evaluation for heavy traffic and transient LAN is proposed. In the method a node on a LAN is modeled as a set of detailed communication protocol models. By parallel and event driven processing of the models, high accuracy and high time-resolution of evaluation of LAN behaviors can be obtained at multi-layer protocols. The LANs at computer education sites have highly loaded peaks, and it is very hard to design large scale educational LANs. Proposed method can be used to evaluate such cases of heavy traffic and transient LAN.

key words: traffic analysis, event driven simulation method, LAN, workstation system

1. Introduction

Performance evaluation of computer communication network has been a significant subject of research in the last decade. At present, computer network traffic consists of various kinds of data from simple text to multimedia hypertext. High accurate traffic estimation and analysis of behaviors are very important for construction of Local Area Network (LAN) at a large scale workstation system.

Three kinds of methods can be used for evaluating the performance of a LAN: (1) by an experiment with existing actual LAN, (2) by an analytical modeling, and (3) by a simulation modeling. An experiment by an actual LAN is not practical for the system consisting of many (about 50 or more) workstations. An analytical modeling [1] [2] is useful for evaluation of LAN that processes low transience load and low traffic, and simple communication protocols. However, at present LAN systems, it is hard to build an analytical model. This is because (i) actual computer communications are based on multiple layer protocols, (ii) at client-server systems, there are some dependencies in the message arrival at the server and clients. Murata *et. al.* [1] built an analytical model of LAN with multiple layer protocols. The model is used for analyzing the behaviors of a client-server system, but the dependencies between a server and clients are not well discussed. Ibe *et. al.* [2] proposed to use of stochastic reward nets to model a client-server system on CSMA/CD and token ring network. However, the behaviors of upper layer protocols than data link layer are not discussed. Furthermore, it is not easy to evaluate LAN with high transience and high-load traffic because of using piecewise averaging. On the other hand, a simulation modeling is useful for transient LANs at such a large scale workstation system [3] at educational institutions.

O'Reilly *et al.* [4] proposed a simulation method based on a grouping and time-slot quantization, and it is not practical for transient LAN. Marino *et al.* [5] proposed a fast discrete event simulation method for CSMA/CD LAN. However, they did not discuss applications of multiple layers. It is not practical to evaluate behaviors of existing actual LANs with single layer protocol model. This is because the actual LANs are based on multiple layer protocols [6] [9] [7] and sometimes depend on man-machine interactions. Interactions between protocol layers have influence on the performance of network. For example, retransmissions on transport layer may cause a congestion on datalink layer, and the congestion may cause more retransmissions on transport layer. Recently there is no paper that gives solutions for such problems. Some efficient commercial network simulators (*e.g.* OPNET) are used widely. However their principles and details are not announced.

In this paper we propose a new modeling and simulation method for traffic analysis and performance evaluation of LAN [8]. Proposed model consists of multiple layer protocols, and the behaviors of each layer are simulated with an event-driven scheme and without time-slot quantization. Therefore the method has high accuracy and highly time resolution for even heavy and transient LAN. In addition to this, the influence of the interactions between protocol layers can be evaluated. In this paper we assume two kinds of CSMA/CD (Ethernet) LAN (10BASE-T: 10 Mbps twisted pair lines, star topology [6]) as physical and data link layer protocols and TCP/IP as transport and network layer ones. However, the method can be used for other protocols and can be applicable for upper layer protocols consisting of man-machine interactions.

Being as a main topic of this study, the multiple layer modeling and simulation method are described in section 2. In additions, we introduce a simple and efficient method to simulate CSMA/CD protocols for

Manuscript received December 5, 1996.

Manuscript revised March 26, 1997.

[†]Department of Information Electronics, Graduate School of Engineering, Nagoya University

[††]Education Center for Information Processing, Nagoya University

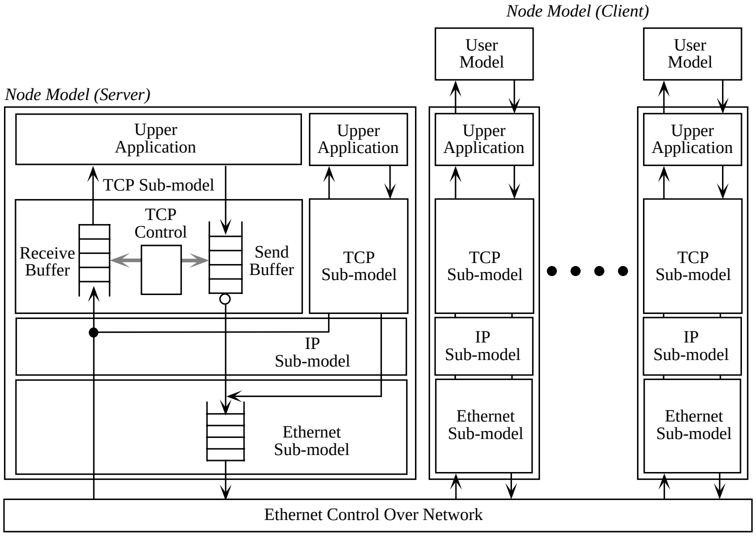

Fig. 1 Overview of the simulation model

LANs that satisfy certain restriction. In section 3, we adapt this method to evaluate a client-server system on a high-transience and high-load LAN. Some experimental results by simulation and an actual system are shown.

2. Simulation Model

2.1 A scheme of the multi-layer modeling

In this method, the behaviors of network protocols are described in detail. As an application of this method we assumed protocols as shown in Table. 1.

By adding an upper application model and an user reaction model to these protocol models, the behaviors of various network applications on various situation can be simulated. A network node is expressed as a set of an user reaction model, an upper application model and protocol sub-models: an Ethernet, IP and TCP sub-models. The Ethernet sub-model is for data link and physical layers that utilize CSMA/CD [6]. TCP and IP sub-models represent each layer protocol. Proposed method has an ability to handle plural sub-models in each layer at a workstation.

Fig. 1 shows an overview of our network model for a client-server computing. In this model, TCP sub-model represents a TCP socket. Therefore, a model of client station has a TCP sub-model and an application model. On the other hand, a model of a server station has TCP sub-models and application models as many as client stations.

Table 1 Protocols assumed on this model

Transport Layer	TCP
Network Layer	IP
Data Link Layer Physical Layer	IEEE 802.3 CSMA/CD Twisted Pair Cable

Simulations on these models are processed by event driven method. Processes on each sub-model are caused by events, which are scheduled by itself or other sub-models. After the process on a sub-model, the sub-model computes next event schedule and current time is revised. On sub-models other than Ethernet one, events are scheduled by sub-models in their own nodes that the models are included. However, events for Ethernet sub-models are scheduled by the state of all Ethernet sub-models on whole network model because of dependencies between Ethernet sub-models.

2.2 Ethernet Sub-Model

2.2.1 State Transition

Ethernet sub-model is presented as a state transition model. In the specifications of IEEE 802.3, CSMA/CD algorithm is defined as a state transition model that includes 11 states. For purpose of efficient simulation and for our Ethernet sub-model we reduced it to six states shown in Fig. 2 from the original state transition model. In spite of this reduction of states, the simulation can be operated without decreasing accuracy.

An Ethernet sub-model of node i ($i = 0, \cdots, N$) [†] has a transmission state $S^{(i)} \in$ {SLP, CS, CR, TR, CD, WT}, a last carrier generation time $t_s^{(i)}$, a sum of inter-frame gap and a last carrier vanishing time $t_e^{(i)}$, and the number of collision in transmission of one frame $n_c^{(i)}$. Where, six states of $S^{(i)}$ are defined in this paper, not in IEEE standard 802.3. A state $M^{(i)}$ of a node i is presented as $M^{(i)} = \{S^{(i)}, t_s^{(i)}, t_e^{(i)}, n_c^{(i)}\}$. Beside this,

[†] The node 0 means a server workstation

Table 2 State transition of Ethernet sub-model

Current State	Event	Action	Next State
SLP	Data exists in send buffer and medium is free	$n_c^{(i)} \leftarrow 0$	CR
	Data exists in send buffer and medium is busy	$n_c^{(i)} \leftarrow 0$	CS
CS	Medium is free	$t_s^{(i)} \leftarrow t, N_c \leftarrow N_c + 1$	CR
CR	Critical period is up without collision	$N_c \leftarrow N_c - 1$	TR
	Critical period is up with collision	$N_c \leftarrow N_c - 1, n_c^{(i)} \leftarrow n_c^{(i)} + 1$	CD
TR	Sending frame done	$t_e^{(i)} \leftarrow t$	SLP
CD	Sending JAM signal done, $n_c^{(i)} = 16$	$t_e^{(i)} \leftarrow t$	SLP
	Sending JAM signal done, $n_c^{(i)} < 16$	$t_e^{(i)} \leftarrow t$, compute $T_b^{(i)}$	WT
WT	Waiting time elapsed and medium is free	$--$	CR
	Waiting time elapsed and medium is busy	$--$	CS

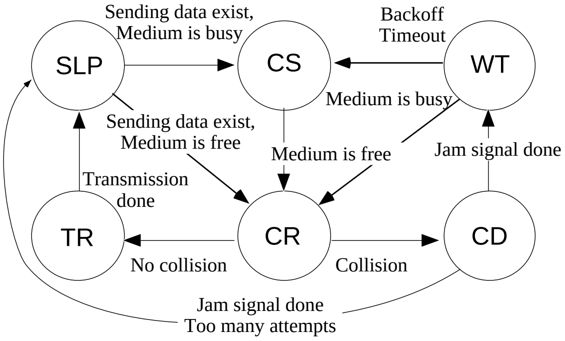

Fig. 2 State transition diagram of Ethernet sub-model

the number of nodes that may detect a collision, N_c is defined.

At a node i, the next event schedule $t_n^{(i)}$ is determined by $M^{(i)}$, the states $M^{(j)}$ of other nodes j $(= 0, 1, \cdots, i-1, i+1, \cdots, N)$ and N_c. When the schedule expires, then the transition state $S^{(i)}$ of the node i is changed and new $t_n^{(i)}$ is computed. The state transition diagram and the state transition table are shown in Fig. 2 and Table 2, respectively.

Six transition states of $S^{(i)}$ are defined and described as follows.

SLP (SLeeP) : A state that the node is not transmitting any data. When a data sending request is received at Ethernet sub-model of the node, the state is transferred. If the medium is busy, the state is transferred to CS, otherwise CR.

CS (Carrier Sense) : Means that the node can not transmit a frame because the medium [†] is busy or the node is waiting for the end of inter-frame gap time. When the medium becomes free and inter-frame gap time elapses, the state is transferred to CR.

CR (CRitical) : In the states of CR and TR, the node is transmitting a frame. CR is a state that a frame sent from the node may occur a collision. In the specification of CSMA/CD algorithm of IEEE 802.3, the period that a node is sending a frame is defined as one state. We introduced these two states to Ethernet sub-model for efficient decision of collision occurrence in simulation.

When a node is in CR, and the node is transmitting a frame, however a carrier generated by the node propagates to not all nodes in the network. Hence, another node can not detect the carrier, and may start transmission. This means that the node may detect a collision. If a node detects a collision, the state of the node is transferred to CD, otherwise TR.

TR (TRansmit) : Means a state that the node should be transmitting a frame successfully. When the transmission is done, the state is transferred to SLP.

CD (Collision Detect) : Means that the node is sending a JAM signal to the medium after a collision detection. After JAM signal is sent, if a number of collision related the sending packet n_c is $n_c < 16$(maximum trial times to send a frame defined by IEEE 802.3), waiting time to retransmission the packet is calculated. Then the state is transferred to WT, or the state is transferred to SLP if $n_c = 16$.

Furthermore, we assume that all nodes detect a carrier when at least one node is in CD. This assumption restricts the scale of simulated network. We describe details of the restriction in 2.2.5

WT (WaiT) : Means that the node is waiting for its random back-off delay to expire before attempting to retransmit a frame. When the delay is expired, if the medium is busy then the state of the node is transferred to CS otherwise CR.

2.2.2 Decision of Carrier Status

As stated above, status of carrier has to be judged when a state transition that the node's previous state is SLP or WT takes place. Assuming that a carrier detection

[†] Medium means physical transmitting line.

will be made at a node i $(= 0, 1, \cdots, N)$, and other node is j $(= 0, 1, \cdots, i-1, i+1, \cdots, N)$. If a node j is in transmitting states $S^{(j)} \in \{\mathrm{TR}, \mathrm{CD}\}$ then a carrier of the node j propagates whole of the medium and the medium is busy. On the other hand, in a case of $S^{(j)} \in \{\mathrm{SLP}, \mathrm{CS}, \mathrm{CR}, \mathrm{WT}\}$, carrier status depends on the time when the carrier is sent and stopped by the node j. If $S^{(j)} = \mathrm{CR}$, the node has transmitted a frame at a time $t_s^{(j)}$ and a carrier of the frame does not reach the node i until $t_s^{(j)} + \tau^{ij}$ because of a propagation delay τ^{ij} between node i and j. On the other hand, even if a node j is not transmitting ($S^{(j)} \in \{\mathrm{SLP}, \mathrm{CS}, \mathrm{WT}\}$), if current time is before $t_e^{(j)} + \tau^{ij}$, the node i observes a carrier generated by the node j. Where $t_e^{(j)}$ is a time the last carrier disappeared.

Therefore, by following condition, we can conclude that a carrier by another node j exists at the node i and the medium must be busy.

$$\text{Channel is busy at node } i = \begin{cases} \exists j \text{ s.t. } S^{(j)} \in \{\mathrm{TR}, \mathrm{CD}\} \\ \qquad \text{or} \\ \exists j \text{ s.t. } \{S^{(j)} = \mathrm{CR} \wedge t_s^{(j)} + \tau^{ij} \leqq t\} \\ \qquad \text{or} \\ \forall j, S^{(j)} \in \{\mathrm{SLP}, \mathrm{CS}, \mathrm{WT}\}, \\ \exists m \text{ s.t. } \{m \neq i; t \leqq t_e^{(i)} + \tau^{mi}\} \end{cases} \tag{1}$$

2.2.3 Decision of Collision Occurrence

With the CSMA/CD algorithm, a collision occurs when a node i has transmitted and another node j begins to transmit without detection of a carrier of the node i. Therefore after a case of all of the nodes is not transmitting (that is to say for all node j, $S^{(j)} \in \{\mathrm{SLP}, \mathrm{CS}, \mathrm{WT}\}$), only during the time beginning first transmission ($t_s^{(k)}$) at a node k and the time $T_c = t_s^{(k)} + \tau_m^{(k)}$ that is all nodes receive the carrier of k, a collision may occur. Where, $\tau_m^{(k)}$ is the maximum propagation delay between node k and another node as: $\tau_m^{(k)} = \max_{j \neq k}\{\tau^{jk}\}$.

When the state is CR, we can decide if a collision is made at $t = T_c$ or not by:

$$\begin{cases} \text{if} \quad N_c > 1 \quad \text{then} & \text{collision} \\ \qquad\qquad\qquad \text{otherwise} & \text{no collision} \end{cases} \tag{2}$$

A time $t_c^{(i)}$ when the node i detects a collision, is given by,

$$t_c^{(i)} := \min_{j \neq i, S^{(j)} = \mathrm{CR}} \{t_s^{(j)} + \tau^{ij}\} \tag{3}$$

Note that this method to decide a collision occurrence is only used under the restriction of the simulated network that described in 2.2.5. It restricts the scale of simulated network to be small. However, networks that satisfy the restriction are used widely, and most of transient networks such a large scale workstation system at educational institution satisfies the restriction. Furthermore, we can decide a collision occurrence easily by using this method under the restriction.

2.2.4 Event Scheduling

If an event occurred on a Ethernet sub-model, next event is scheduled for it as follows. The entry of items represents the state of the node i after state transition. $t_n^{(i)}$ is the schedule of next event at node i.

SLP : When $S^{(i)} = \mathrm{SLP}$, the next event schedule $t_n^{(i)}$ can not be computed in the process of the Ethernet sub-model. Because, existence of data in an Ethernet sending buffer is affected by the result of a transport sub-model processed before Ethernet sub-model.

$$t_n^{(i)} := \infty \tag{4}$$

CS : The next event is detection of no carrier at the node. If a node i such that $S^{(i)} = \mathrm{CS}$ detects the carrier from a node j such that $S^{(j)} \in \{\mathrm{CR}, \mathrm{TR}, \mathrm{CD}\}$, the node i detects a carrier absence after state transition of the node j. On the other hand, if a node i detects the carrier from a node j such that $S^{(j)} \in \{\mathrm{SLP}, \mathrm{CS}, \mathrm{WT}\}$, the node i detects a carrier absence from the node j at $t = t_e^{(j)} + \tau^{ij}$. Thus,

$$t_n^{(i)} := \begin{cases} \infty \;\; (\exists j \text{s.t.}[S^{(j)} \in \{\mathrm{TR}, \mathrm{CD}\}] \\ \qquad \vee [S^{(j)} = \mathrm{CR} \wedge t < t_s^{(j)} + \tau^{ij}]) \\ \\ \max_{j \neq i}\{t_e^{(j)} + \tau^{ij}\} \;\; (\text{otherwise}) \end{cases} \tag{5}$$

If $S^{(i)} = \mathrm{CS}$, $t_n^{(i)}$ of the node must be computed when at least one $S^{(j)} (j \neq i)$ is transferred between $\{\mathrm{CR}, \mathrm{TR}, \mathrm{CD}\}$ and $\{\mathrm{SLP}, \mathrm{CS}, \mathrm{WT}\}$. Because when $S^{(i)} = \mathrm{CS}$, $t_n^{(i)}$ is depend on the states of other nodes. Otherwise, $t_n^{(i)}$ is computed only when the state of the node itself, $S^{(i)}$ is changed.

CR : The next event is the end time of the period that the node may occur a collision.

$$t_n^{(i)} := T_c = t_s^{(k)} + \tau_m^{(k)} \tag{6}$$

TR : The next event is the end of the transmission of a frame.

$$t_n^{(i)} := t_s^{(i)} + \tau_p^{(i)} \tag{7}$$

Where, $\tau_p^{(i)}$ is a sum of a time required for the node to transmit a frame and an inter-frame gap.

CD : The next event is the end time of transmission of a JAM signal.

$$t_n^{(i)} := t_c^{(i)} + \tau_{jam} \tag{8}$$

Where, $t_c^{(i)}$ is the time when the node i detected a collision, τ_{jam} is a sum of a time of a JAM signal and an inter-frame gap.

WT : The next event is that the node attempts to retransmit a frame.

$$t_n^{(i)} := t_c^{(i)} + \tau_{jam} + \tau_b^{(i)} \tag{9}$$

Where, $\tau_b^{(i)}$ is a random back-off delay computed on the transition from CD to WT.

2.2.5 Restriction of Ethernet Sub-Model

As we mentioned before, to utilize the method to decide the carrier occurrence described in 2.2.3 the simulated network has to satisfy the following restriction. When a node i comes in a collision and $t_c^{(i)} < T_c$, the node i sends a part of JAM signal in CR and rest of it in CD after state transition at $t = T_c$. However, if

$$t_c^{(i)} + \tau_{jam} < T_c \tag{10}$$

then, the node i finishes the transmission in CR. This condition contradicts the definition of CR in which node i transmits a part of frame and has not finished transmission yet. For this reason, proposed model restricts for networks simulated to satisfy following condition.

$$\max_{i \neq j}\{\tau^{ij}\} - \min_{i \neq j}\{\tau^{ij}\} < \tau_{jam} \tag{11}$$

Furthermore, we assumed that all nodes detect a carrier when at least one node such that $S^{(i)} = \text{CD}$ exists. That is to say, JAM signal sent from the last node that detects a collision has to propagate to all nodes before a node detect the end of JAM signal from other nodes. The worst case of this condition is that packets from two nodes a, b that propagation delay between them is most short collide with each other, and a packet from another node c that propagation delay between a and c is most long collides with the packet from a. Then the first node that detects collision is a or b, and the last one is c. The minimum of the time when a node detects the end of JAM signal is $t_s^{(a)} + 2\tau^{ab} + \tau_{jam}$. The time when the JAM signal from the node c propagates to all nodes is $t_s^{(a)} + \tau^{ac} + \max_{j \neq c}\{\tau^{cj}\}$. Hence the simulated network have to satisfy following condition.

$$2\min_{i \neq j}\{\tau^{ij}\} + \tau_{jam} > 2\max_{i \neq j}\{\tau^{ij}\} \tag{12}$$

2.3 IP sub-model

The functions of IP are routing, fragmentation and reassembling of IP datagrams. We assumed that the simulated network consists only one logical network. So the operation according to routing can be ignored. Furthermore, we assumed that TCP offers no IP fragmentation. As a result of this, the effect of IP to message transmission time is only the format of IP datagram. So we modeled IP as a simple function that adds an IP header length to a TCP segment, and removes an IP header length from messages from Ethernet sub-model.

2.4 TCP sub-model

TCP offers reliable full duplex channels on unreliable protocols (*e.g.* IP). To achieve reliable data transfer on unreliable protocols, TCP uses a sliding window mechanism. This involves the use for data sequence numbers, acknowledgment messages, sending and receiving windows, and retransmission.

The performance of a TCP connection depends on various policies employed by the TCP entities, regarding transmission, retransmission time-out adjustment, window size, etc. We note that while protocol standards provide specifications of functionality, message formats and semantics, they do not specify policies regarding performance. Consequently, implementations that conform to the standards can and do exhibit significant variation in their performance. So, we selected one implementation of TCP based on UNIX 4.3 BSD [10] from many ones for our model. We assumed the parameter of the TCP system configuration as follows.

Sending Buffer Size: 4,096 octet.

Receiving Buffer Size: 4,096 octet.

Maximum Segment Size: 1,460 octet. This value does not demand IP fragmentation on IEEE 802.3 network.

Retransmission Time-out Algorithm: First we compute smoothed round-trip time T_{srtt_n} from previous one $T_{srtt_{n-1}}$ and measured round trip time T_{rtt}.

$$T_{srtt_n} := 0.9T_{srtt_{n-1}} + 0.1T_{rtt}$$

Retransmission time-out T_{rto} for next transmission is computed as follows.

$$T_{rto} := 2T_{srtt_n}$$

T_{rto} [sec] is limited as $1 \leqq T_{rto} \leqq 64.0$. When the TCP socket retransmitting segments, the evaluation of T_{srtt_n} is not processed. The retransmission time-out for retransmitted segment is computed by Karn's algorithm [11].

Maximum Retransmittion Times: Infinite.

In this simulation model some parts consisting of segment header (source, destination, sequence number of segment, ACK, window size and data length) are exchanged between two TCP sub-models. Some TCP processes required to implement the sliding window mechanism are performed at TCP sub-models. Events that cause the processes of a TCP model are receiving send request, receiving a segment, expiration of persist timer, expiration of retransmit timer, expansion of free space of receiving buffer. If a sending request is generated as a result of processes in TCP sub-model, then a sending request is passed to Ethernet sub-model. We did not consider the CPU processing time for TCP processes in this model.

3. Experiments of Simulation

In this section, some simulation results obtained by proposed modeling and simulation method are shown.

3.1 Simulation Conditions for Transient-Traffic LAN

A large scale workstation system [3] consisting of a file server and n client workstations for computer education is chosen to illustrate the simulation of a heavy and transient traffic LAN. In a class room of computer education that uses such a system, most of students begin to operate workstations all together directly after an instruction for students to operate a file on the file server. As a result of this, network requests from client workstations used by students concentrate on the file server, and a high load and transience are instantly made on the LAN.

File transfer from a file server to client workstations on a 10 BASE-T (10 Mbps, Star) Ethernet LAN with a HUB shown in Fig. 3 is selected as an example of such a situation. All client workstations start to send request messages to the server. The server workstation sends reply message to clients after receiving the request messages. We assumed that the time when a client workstation i starts to send request messages, $t^{(i)}_{start}$, is distributed according to the time required for students to response to the instruction. Complete time $t^{(i)}_{end}$ may vary. The probability density function of $t^{(i)}_{start}$, $f(t_{start})$ is determined as follows.

$$f(t_{start}) = \begin{cases} \dfrac{\exp(\frac{-(t_{start}-\mu)^2}{2\sigma^2})}{\int_0^\infty \exp(\frac{-(\xi-\mu)^2}{2\sigma^2})d\xi} & (t_{start} \geq 0) \\ 0 & (t_{start} < 0) \end{cases} \qquad (13)$$

Eq. (13) indicates a distribution such that the probability of random number $t_{start} < 0$, is zero for the normal distribution $N(\mu, \sigma^2)$. Random values $t^{(i)}_{start}(i = 1, 2, \cdots, N)$ generated from $f(t_{start})$ define behaviors of users' interactions at the client workstations, and are used as simulation parameters. Parameters μ and σ give the degree of concentration of network accesses. Three access concentration models are developed for representation of various command input methods (Table 3). *Model M, K* and *O* correspond to mouse operation, keyboard operation and at-a-time for comparison respectively. The numerical values of parameter μ, σ of the models are referred previous experimental results with an existing system and in an actual class consisting of 146 students obtained by Ishihara *et al.* [12]

Table 3 Parameters of access concentration model

Model	μ sec	σ sec
Model M (mouse operation)	2.2	1.6
Model K (keyboard operation)	6.3	10.6
Model O (at-a-time)	1.0	0.0

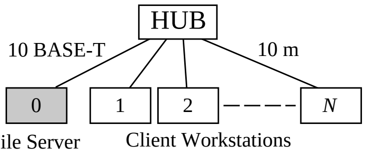

Fig. 3 Simulated network

3.2 Application Model

As an example of an application that utilizes TCP, we modeled a simple file transfer program. The operation flow of this program model is mentioned below.

1. A client program sends a request message (20 bytes) to server.
2. Server program sends the requested data stream to the client via TCP by the size of sending buffer of the server program S_{server}.
3. Client program receives data from the TCP receiving buffer by own receiving buffer of R_{client} bytes. The time to required to read the TCP receiving buffer is assumed as an exponentially distributed random variable. The mean of the distribution is defined as 0.7 msec for reading 1024 bytes. This value is defined from experimental results on an actual system.

We assumed that virtual connections between clients and server are established before the client program sends a request message. CPU processing time on server station is neglected.

The Ethernet frame sizes of messages which include the header of TCP, IP and Ethernet are as follows.

- Request Message from client: 74 bytes.
- Data Stream from Server: min{Maximum segment size of TCP, S} + 58 bytes or smaller. This value

varies according to the free space of receiving buffer of client TCP.

- Acknowledgment Message from Client: 64 bytes.

3.3 Simulation Results

Some experimental results are shown here. Assuming N is the number of client workstations.

First, we show the simulationn result when $N=1$. Fig. 4 shows the file transfer time $T_{trs}^{(1)} = t_{end}^{(1)} - t_{start}^{(1)}$ versus the sending buffer size of the server program S_{server}. The receiving buffer size of a client program R_{client} is fixed to 4096 bytes. Experimental results with a 10BASE-5 (10 Mbps, coaxial bus) actual workstation LAN[3] are also presented for comparison. This experimental result includes an overhead to connect TCP connection between the server and a client. The network topology of the actual system is differ from the simulated network in the physical layer. However, logical topology of the network is same as simulated network. It satisfies the restriction of the simulated network described in 2.2.5. When $S_{server} \geqq 512$bytes, the simulation result agrees with experimental results. When S_{server} is small, the simulationn error increases. The small sending buffer size causes many small packets and many reading process on client. As the result of this, the errors in reading process accumulates and enlarges the total simulation error. We think main causes of errors are the modeling method of reading process from TCP receiving buffer and the parameter.

Fig. 5 presents the averaged file transfer time $\overline{T_{trs}}$ versus the number of client workstations N. $\overline{T_{trs}}$ is defined by

$$\overline{T_{trs}} = \overline{t_{end}^{(i)} - t_{start}^{(i)}} = \frac{1}{N}\sum_{i=1}^{N} T_{trs}^{(i)} \qquad (14)$$

The size of transferred file is 1 Mbytes. S_{server} and R_{client} are both 1024 bytes. The calculated values are averages of some simulation trials. The numbers of trials of simulations are determined by N as shown in Table 4. In the actual experiments, $t_{start}^{(i)}$ is given by the same driver program as simulation experiments.

At the model M, file transfer time $\overline{T_{trs}}$ increases in proportion to linearly with N. On the other hand, at the model K, $\overline{T_{trs}}$ keeps low values until $N=20$, and $\overline{T_{trs}}$ increases rapidly after $N=20$. The experimental results of the simulation are analogous to ones by existing system, but a simulation error increases at $N>20$. We think that the reasons of the errors are ignoring processing time of server station and overhead of TCP connection. Because these values increase with N. But we conclude the simulation results are very useful to evaluate the behavior of networks. However, these results are sufficient to know characteristics of such a high-load LAN.

Table 4 Number of trial for averaging

N	Number of trial
$1\ldots25$	20
$30, 35$	15
$40\ldots75$	10
100	5

Fig. 6 presents the averaged collision rate $\bar{c}$ versus N. $\bar{c}$ is defined by

$$\bar{c} = \frac{1}{N+1}\sum_{i=0}^{N}\frac{n_c^{(i)}}{n_a^{(i)}} \qquad (15)$$

where $n_a^{(i)}$ and $n_c^{(i)}$ are the trial number of transmitting at an Ethernet sub-model of a node i, and the number of collision respectively. Note that a case of $i=0$ means the file server workstation. In addition to this, Fig. 6 represents the averaged retransmission count of TCP per a connection $\bar{r}$. $\bar{c}$ increases rapidly at low values of N. $\bar{r}$ increases rapidly when $\bar{c} > 0.3$. Furthermore, $\overline{T_{trs}}$ in Fig. 5 increases with $\bar{r}$. Note that $\bar{r}$ of model M is larger than one of model O. It means that adaptive control of retransmission timeout of TCP does not follow to rapid increase of traffic because of the distribution of t_{start}. However, the increasing rate of $\overline{T_{trs}}$ and $\bar{r}$ when $\bar{c} > 0.3$ is approximately linear, and the failure of network because of the multiplier effect of collision on Ethernet and retransmisions of TCP is not observed in this simulated situation.

These results lead us to the conclusion that guidelines for the number of workstations at educational LAN that keeps ideal low file transfer time and low collision rate are 15 or 20 nodes for 10 Mbps LAN. However, the failure of network does not occur at least 100 nodes.

The execution time of the simulation depends on N, access concentration model, channel speed, and transmitted file size. With conditions of $N=20$, Model M for access concentration model, channel speed of 10 Mbps, and transmitted file size of 1 MB, it required 39 sec turnaround for a trial of the simulation. The simulation experiments are performed on SPARC center 2000.

4. Conclusions

A new method of modeling and simulation for high accurate performance analysis of LAN is proposed. In the method network node models based on multi-layer communication protocols are engaged in parallel with an event driven scheme. Then, the method is able to analyze a network traffic high accurately even if the traffic load of network is high and the load has high transience. For efficiency simulation, we introduced a simple method to decide a collision occurrence in CSMA/CD algorithm without decreasing accuracy. However, the method is only used for networks that

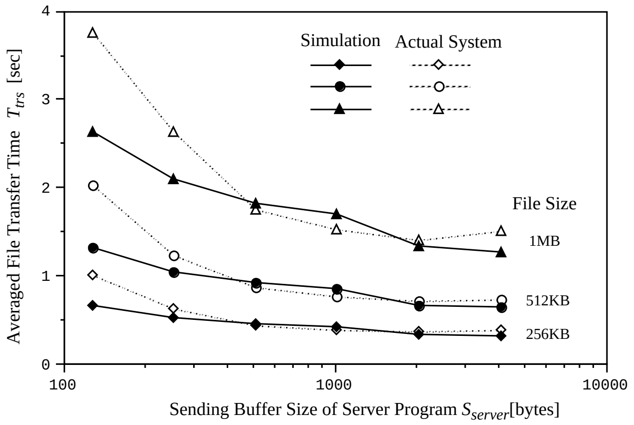

Fig. 4 Averaged file transfer time versus sending buffer size of server program.

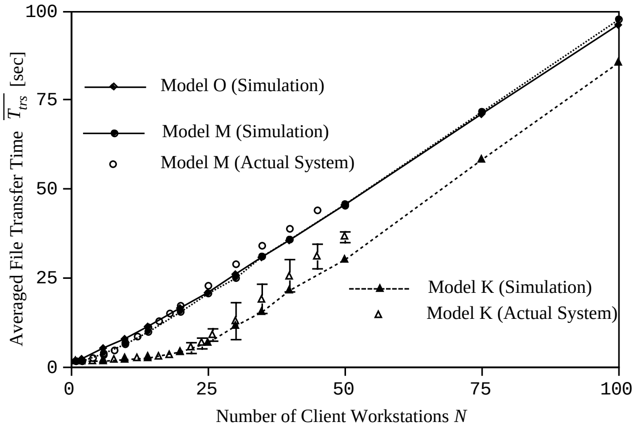

Fig. 5 Averaged file transfer time versus the number of client workstations.

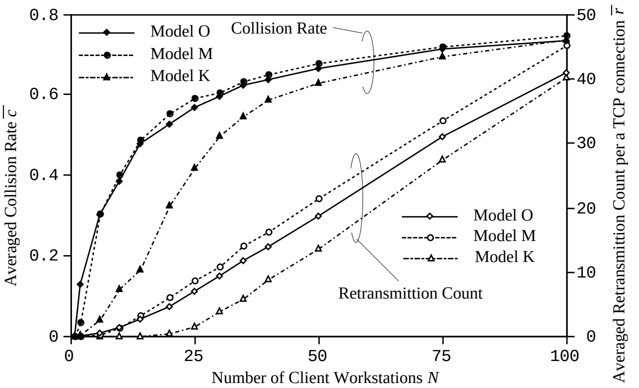

Fig. 6 Averaged collision rate versus the number of client workstations.

satisfy a condition to restrict the scale of the networks. The accuracy is confirmed with results of experiments by an actual existing workstation LAN.

The method was applied for a case of heavy and transient traffic LAN at an educational workstation system. By the experiments, valuable details of behaviors of the LAN which can not be obtained by other single layer simulation models are obtained.

Modeling for multiple networks consisting of bridges and/or routers, applications to variety of protocols are topics for future study. Furthermore we are planning to utilize this method to analyze large scale distributed file systems used for an educational LAN.

References

[1] M. Murata, and H. Takagi, "Two-layer modeling for local area networks," IEEE Trans. Commun., vol.36, no.9, pp.1022–1034, 1988.

[2] O. C. Ibe and H. Choi and K. S. Trivedi, "Performance evaluation of client-server systems," IEEE Trans. Parallel and Distributed Systems, vol.4, No.11, pp.1217–1229, 1993.

[3] M. Okada, K. Sakurai and A. Iwata, "Implementation of large scale distributed workstation system for practice on computer science, " Trans. IPS. Japan, vol.37, no.12, pp. 2447–2456, 1996

[4] P. J. P. O'Reilly and J. L. Hammond, Jr., "An efficient simulation technique for performance studies of CSMA/CD local networks," IEEE J. Sel. Areas Commun., vol.SAC-2, No.1, pp.238–249, 1984.

[5] P. Marino and A. del. Rio, "An accurate and fast CSMA/CD simulator," Microprocessing and Microprogramming, vol.39, pp.197–190, 1993.

[6] ANSI/IEEE Standard 802.3, "Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications," IEEE Press, 1993.

[7] J. Postel, "Transmission Control Protocol," RFC 793, Information Science Institute / University of Southern California, 1981.

[8] S. Ishihara, M. Okada, A. Iwata and K. Sakurai, "Network traffic model with event driven method on local area network," IEICE Trans. Fundamentals, vol.J78–A, no.8, pp.961–964, 1996 (in Japanese).

[9] J. Postel, "Internet Protocol Specification," RFC 791, Information Science Institute / University of Southern California, 1981.

[10] S. J. Leffler, M. K. McKusick, M. J. Karels and J. S. Quarterman, "The Design and Implementation of the 4.3BSD UNIX Operating System," Addison-Wesley Publ., 1989.

[11] D. E. Comer, d. L. Stevens, "Internetworking with TCP/IP Vol. II: Design, Implementation, and Internals," Prentice-Hall, Inc., 1991.

[12] S. Ishihara, M. Okada, A. Iwata and K. Sakurai, "Traffic analysis on the LAN for education," Proc. of Symposium on Informatics 1996, pp.17–23, 1996 (in Japanese).

Susumu Ishihara received his BSEE and MSEE degrees from Nagoya University in 1994 and 1996. He is currently studying as a candidate for doctorate at Department of Information Electronics, Graduate School of Engineering, Nagoya University. He received an incentive of 1994 Tokai-Selection joint conference of the six institutes of electrical and related engineers. His research interests are in the design and analysis of local area network and distributed workstation systems.

Minoru Okada received his BSEE and Ph.D. degrees in 1984 and 1992 from Nagoya Institute of Technology, Japan and Nagoya University, Japan, respectively. From 1988 to 1990 he was a Special Researcher of JSPS fellowship. Since 1990, he has been with Nagoya University, where he is an Associate Professor in the Education Center for Information Processing. During 1995–1996, he was a Visiting Associate Professor in the Department of Electrical and Computer Engineering at the University of Michigan, on leave from Nagoya University. His research interests are in the areas of image recognition and understanding, computer graphics, their application to medical diagnosis, and large scale distributed computer systems. He is a member of IPS Japan, and IEEE.