Setting Target_Rotation_Times in an 802.4 Network

Mangala Gorur
Alfred C. Weaver

Computer Science Report No. TR-86-21
September 18, 1986

Setting Target_Rotation_Times in an 802.4 Network

Mangala Gorur
Alfred C. Weaver
Dept. of Computer Science
University of Virginia
Charlottesville, Va 22903

The IEEE 802.4 token bus protocol for local area networks is emerging as a popular standard for factory automation applications. The enormous interest in this protocol is due to its various distinguishing features, one of which is the provision of a prioritized mechanism to access the transmission medium. The implementation of the priority scheme by any station in an 802.4 network is optional. If a station implements the priority scheme, then the objective is to allocate bandwidth to lower priority frames only after transmission of higher priority frames. Each of the lower three access_classes is assigned a target token rotation time. This goal rotation time is different for each access_class and is known as the *Target_Rotation_Time* (TRT) of an access_class. The TRT setting at each access_class decides the amount of time available to serve messages of that priority. The IEEE 802.4 token bus standard specifies only the maximum values for the Target_Rotation_Times. Hence it is essential to know how to set the TRTs to achieve a desired priority scheme.

In this paper we present an analytic model which can be used to solve for the TRT settings which will implement a user-defined priority scheme. For example, suppose that the user wants normal service at Time_Available access_class until network throughput rises to α, with $0 \le \alpha \le 1$. Then letting N be the number of stations in the network, X_T be the transmission time of a token, X_M be the transmission time of a message, and λ_s, λ_{ua}, λ_{na}, and λ_{ta} be the arrival rate in packets per second at the Synchronous, Urgent_Asynchronous, Normal_Asynchronous, and Time_Available access_classes respectively, and using the analytic model, the effective value of the TRT_{ta} is,

$$eff\,(TRT_{ta}) = \frac{N \cdot X_T}{1 - N \cdot X_M \cdot (\lambda_s + \lambda_{ua} + \lambda_{na} + \lambda_{ta})}.$$

The effective value can be higher than the actual TRT setting by up to one message transmission time. This is according to the specifications of the standard that the transmission of a message once started will go on to completion even if it runs past the expiration of the timers.

The analytic predictions show close agreement with the simulation results. Thus, given a user-defined performance target (eg., "Time_Available access_class to receive normal service until network throughput exceeds α"), we can calculate the TRT settings which will achieve that goal.

1. Introduction

The 802.4 token passing access method offers four levels of service called *access_classes* and messages can be transmitted at any of the four priorities. The four access_classes in descending order of priority are Synchronous access_class, Urgent_Asynchronous access_class, Normal_Asynchronous access_class, and Time_Available access_class. The implementation of the priority scheme by any station is optional. By suitably setting the variables associated with the implementation of the priority scheme, it can be ensured that this scheme gives preference to frames of higher priority. The amount of time available to transmit messages from the Synchronous access_class is decided by the value of the variable High_Priority_Token_Hold_Time (HPTHT). The amount of time available to transmit messages from each of the lower three access_classes is decided by the value of the Target_Rotation_Time (TRT) setting at each access_class. This setting is different for each access_class.

The TRT setting at each access_class limits the token cycle time at each access_class. In this paper we present an analytic model that can be used to calculate the values of TRT settings to obtain optimum service at an access_class until a user-defined throughput is reached.

1.1. An analytic model for determining TRTs

A problem faced by token bus network designers and operators is the selection of Target_Rotation_Times which will implement a desired priority scheme. It is possible to determine the individual TRT setting of an access_class so as to obtain maximum possible service at an access_class (that is, on the average a message gets transmitted within one token cycle) at an access_class until network throughput reaches or exceeds a user-defined threshold. For purposes of analysis presented in this paper, throughput has been defined as the number of data bits transmitted (including all address and framing bits) per bit time expressed as a fraction of the bus capacity.

1.1.1. Definitions

An active access_class at a station is termed a server. The following notation is used:

X_T ≡ duration of a token transmission (seconds/token transmission).

s ≡ the Synchronous access_class.

ua ≡ the Urgent_Asynchronous access_class.

na ≡ the Normal_Asynchronous access_class.

ta ≡ the Time_Available access_class.

λ_{ac} ≡ the mean message arrival rate at each server of an access_class, where ac is s, ua, na or ta.

S ≡ the set of all *Synchronous access_class* servers.

UA ≡ the set of all *Urgent_Asynchronous access_class* servers.

NA ≡ the set of all *Normal_Asynchronous access_class* servers.

TA ≡ the set of all *Time_Available access_class* servers.

R ≡ set of all distinct servers on the logical ring ≡ (S, UA, NA, TA).

λ_x ≡ the mean message arrival rate at server $x \varepsilon R$ (messages/second).

$HPTHT$ ≡ the *High_Priority_Token_Hold_Time*.

TRT_x ≡ the *Target_Rotation_Time* at server $x \varepsilon (UA, NA, TA)$.

TRT_{asy} ≡ the *Target_Rotation_Time* at each server of an access_class, where asy is ua, na, or ta.

$TCT_{x,i}$ ≡ the time from the end of the $(i-1)^{st}$ service period until the beginning of the i^{th} service period (seconds)
≡ token circulation time as seen by server x on the i^{th} token cycle, i.e., the time interval for which the token is away from x.

$TC_{x,i}$ ≡ time from the end of the $(i-1)^{st}$ service period until the end of the i^{th} service period (seconds).
≡ token cycle time as seen by server x on the i^{th} token cycle.

$A_{x,i} \equiv$ the number of message arrivals at x during the interval from the end of the $(i-1)^{st}$ service period until the end of the i^{th} service period.

$Q_{x,i} \equiv$ the number of messages enqueued at server x after the $(i-1)^{st}$ service period.

$f(n) \equiv$ time required to transmit n messages (octet_times).

$t \equiv$ residual time in the token_hold_timer (octet_times).

$eff(t) \equiv$ the effective amount of time which a server can transmit messages.

$= max(0, t + f(1) -$ one $octet\_time)$ octet_times.

$TS_{x,i} \equiv$ duration of the i^{th} service period.

$\overline{TS}_{ac} \equiv$ average service time available to every server of an access_class, where ac is s, ua, na or ta.

1.1.2. Assumptions

In the following section, an analytic expression for the token cycle time of an 802.4 network is derived. The derivation is based on the following assumptions about the characteristics of the network:

1) The protocol management overhead is assumed to be negligible

2) Stations are permanent members of the logical ring.

3) Each station has traffic at all four access_classes.

4) Message arrival at each station follows a Poisson process.

5) Messages from all the servers are of constant length l_M bits and take X_M seconds for transmission.

6) Token is of length 112 bits for a 10 Mbps bus and takes X_T seconds for transmission.

7) All Synchronous access_class servers have the same HPTHT setting.

8) Each station has HPTHT set to the maximum allowed value (52.43 msec for a 10 Mbps bus). Hence messages from the Synchronous access_class normally are transmitted on the same token cycle as they arrive.

9) All servers of the same priority have identical traffic.

10) All servers of the same priority have the same TRT settings.

11) The TRTs are set in the order $TRT_{ua} > TRT_{na} > TRT_{ta}$.

12) Each active access_class has N servers.

1.1.3. Service time

The amount of time available to any server on any token cycle is limited. A station starts transmitting messages when there is residual time in the token_hold_timer. If x is a synchronous server then the average service time of x can be expressed as

$$\overline{TS}_x = \min(eff\,(HPTHT), f\,(\overline{Q}_x + \overline{A}_x\,)). \tag{1}$$

where $\overline{Q}_x$ is the average queue length at x, and $\overline{A}_x$ is the average number of arrivals at x during an average token cycle. If x is an asynchronous server then the average service time of server x can be expressed as

$$\overline{TS}_x = \begin{cases} \min(eff\,(TRT_x - \overline{TCT}_x\,), f\,(\overline{Q}_x + \overline{A}_x\,)) & \text{for} \quad \overline{TCT}_x < TRT_x \\ 0 & \text{for} \quad \overline{TCT}_x \geq TRT_x \end{cases} \tag{2}$$

where $\overline{TCT}_x,\ x\,\varepsilon TA$ is the average token circulation time at every server at the Time_Available class, $\overline{Q}_x$ is the average queue length at x, and $\overline{A}_x$ is the average number of arrivals at x. Refer to [Gorur 86] for a derivation of token cycle time and token circulation time. Assuming an average arrival rate of messages λ_x at server x, the number of messages that have arrived while

a) the token is circulating around the ring and

b) x is being served on the i^{th} token cycle is

$$A_{x,i} \;=\; \lambda_x \cdot TCT_{x,i} + \lambda_x \cdot TS_{x,i} \;=\; \lambda_x \cdot TC_{x,i}. \tag{3}$$

Hence the average number of messages that have arrived at any server x, $x\ \varepsilon\ (S, UA, NA, TA)$, during an average token cycle can be expressed as

$$\overline{A}_x = \lambda_x \cdot \overline{TC}. \tag{4}$$

The time taken to transmit the average number of messages that have arrived at a server x can be expressed as $X_M \cdot \lambda_x \cdot \overline{TC}$, from equation (4). As long as the time to transmit all the messages that have arrived is less than or equal to HPTHT (if x is a synchronous server), or less than or equal to $(TRT_{asy} - \overline{TCT})$ if x is an asynchronous server, the average service time at server x can be expressed as

$$\overline{TS}_x = f\,(\overline{Q}_x + \overline{A}_x). \tag{5}$$

This is true for $\overline{TC} \leq HPTHT$ and $\overline{TC} \leq TRT_{asy}$. In the region $\overline{TC} \leq HPTHT$ it can be assumed that all the messages from the Synchronous access_class get transmitted during the same token cycle. Also in the region $\overline{TC} \leq TRT_{asy}$ it can be assumed that all the messages from that access_class and from access_classes of higher priority that have arrived during a particular token cycle get transmitted during the same token cycle. Hence $\overline{Q}_x$ can be considered to be negligible. Eliminating $\overline{Q}_x$ from equation (5) and substituting the value of $\overline{A}_x$ from equation (4), the average service time can be expressed as

$$\overline{TS}_x = \overline{TS}_{ac} = X_m \cdot \lambda_{ac} \cdot \overline{TC}. \tag{6}$$

Traffic from each access_class gets served in the order of priority, starting from the Synchronous access_class, and if time is available, lower access_classes also get service. The average service time at each of the lower three access_classes can increase with an increase in offered load from that access_class until the average token cycle time equals the TRT of that access_class. Thus the throughput which results in a token cycle time of TRT for an access_class corresponds to the throughput until which maximum possible service can be obtained at that access_class.

1.1.4. Determining TRTs

If the designer determines that the Time_Available access_class should receive maximum possible service until network throughput reaches α, 0≤α≤1, then

$$\alpha = \frac{N \cdot \Lambda_\alpha \cdot l_M}{C \cdot \overline{TC}_\alpha} \tag{7}$$

where Λ_α is the mean number of messages transmitted per station per token cycle at a throughput of α, and $\overline{TC}_\alpha$ is the average token cycle time at a throughput of α. Solving for Λ_α,

$$\Lambda_\alpha = \frac{\alpha \cdot C \cdot \overline{TC}_\alpha}{N \cdot l_M}. \tag{8}$$

From the discussion in section 1.2.3, it follows that maximum possible service at an access_class can be obtained in the region where $\overline{TC} \leq TRT_{asy}$. Hence the average number of messages that arrive during the time interval $\overline{TC}_\alpha$ (region where $\overline{TC}_\alpha \leq TRT_{ta}$) should equal the mean number of messages transmitted in this time interval, i.e., Λ_α. Hence

$$\Lambda_\alpha = (\lambda_s + \lambda_{ua} + \lambda_{na} + \lambda_{ta}) \cdot \overline{TC}_\alpha. \tag{9}$$

Substituting for Λ_α in equation (8) we obtain the sum of λs to be

$$\lambda_s + \lambda_{ua} + \lambda_{na} + \lambda_{ta} = \frac{\alpha \cdot C}{N \cdot l_M}. \tag{10}$$

Many combinations of individual message arrival rates will yield the unique sum satisfying the above equation. Since the token cycle time can be expressed as

$$\overline{TC}_\alpha = N \cdot X_T + N \cdot X_M \cdot \Lambda_\alpha, \tag{11}$$

substituting for Λ_α from equation (9) yields

$$\overline{TC}_\alpha = N \cdot X_T + N \cdot X_M \cdot (\lambda_s + \lambda_{ua} + \lambda_{na} + \lambda_{ta}) \cdot \overline{TC}_\alpha. \tag{12}$$

Since the Time_Available access_class should receive maximum possible service until throughput reaches α, it follows from the discussion of service time in the previous section that the token cycle time at this throughput decides the TRT of the Time_Available access_class. Hence solving for $eff(TRT_{ta}) = \overline{TC}_\alpha$,

$$eff(TRT_{ta}) = \frac{N \cdot X_T}{1 - N \cdot X_M \cdot (\lambda_s + \lambda_{ua} + \lambda_{na} + \lambda_{ta})} \tag{13}$$

which is the effective value of the Time_Available *Target_Rotation_Time*.

As the offered load increases, increasing the network throughput beyond α, the token cycle time increases beyond the TRT of the Time_Available access_class. Hence service to the Time_Available access_class class gets reduced and eventually drops to zero when TCT_{ta} equals TRT_{ta} as given by equation (2). Hence the traffic from the Time_Available access_class does not contribute to the network throughput anymore. Hence the load carried by the network is contributed by the upper three classes alone.

If the designer determines that the service at the Normal_Asynchronous access_class should reach a maximum when the network throughput reaches β, $0 \le \alpha < \beta \le 1$, then

$$\beta = \frac{N \cdot \Lambda_\beta \cdot l_M}{C \cdot \overline{TC}_\beta} \tag{14}$$

where Λ_β is the mean total number of messages transmitted per station per token cycle at a throughput of β. Solving for Λ_β,

$$\Lambda_\beta = \frac{\beta \cdot C \cdot \overline{TC}_\beta}{N \cdot l_M}. \tag{15}$$

The mean number of message arrivals during the time interval $\overline{TC}_\beta$ should equal the mean number of messages transmitted during that time interval, i.e., Λ_β. Hence

$$\Lambda_\beta = (\lambda_s + \lambda_{ua} + \lambda_{na}) \cdot \overline{TC}_\beta. \tag{16}$$

Substituting for Λ_β in equation (15) we obtain the sum of message arrival rates of the upper three access_classes to be

$$\lambda_s + \lambda_{ua} + \lambda_{na} = \frac{\beta \cdot C}{N \cdot l_M}. \tag{17}$$

Of course, many combinations of individual message arrival rates will yield the unique sum in equation (17). The token cycle time in this region can be expressed as

$$\overline{TC}_\beta = N \cdot X_T + N \cdot X_M \cdot \Lambda_\beta. \tag{18}$$

Substituting for Λ_β from equation (16) yields

$$\overline{TC}_\beta = N \cdot X_T + N \cdot X_M \cdot (\lambda_s + \lambda_{ua} + \lambda_{na}) \cdot \overline{TC}_\beta. \tag{19}$$

Since service at the Normal_Asynchronous access_class should be maximum at a throughput of β, it follows from the discussion of service time in section 1.2.3 that the token cycle time at this carried load decides the TRT of the Normal_Asynchronous access_class. Solving for $eff\ (TRT_{na}) = \overline{TC}_\beta$,

$$eff\,(TRT_{na}) = \frac{N \cdot X_T}{1 - N \cdot X_M \cdot (\lambda_s + \lambda_{ua} + \lambda_{na})} \tag{20}$$

which is the effective value of the Normal_Asynchronous *Target_Rotation_Time*.

As the offered load increases, increasing the network throughput beyond β, the token cycle time increases beyond the TRT of the Normal_Asynchronous access_class. Hence service to the Normal_Asynchronous class gets reduced and eventually drops to zero when TCT_{na} equals TRT_{na} as given by equation (2). At this point traffic from the Normal_Asynchronous access_class does not contribute to the network throughput anymore, so the load carried by the network is contributed only by the upper two classes.

If a designer determines that the service to the Urgent_Asynchronous access_class should be maximized when the network throughput is γ where $0 \le \alpha < \beta < \gamma \le 1$, then

$$\gamma = \frac{N \cdot \Lambda_\gamma \cdot l_M}{C \cdot \overline{TC}_\gamma}. \tag{21}$$

Using logic similar to the previous discussion, the number of messages transmitted during the time interval $\overline{TC}_\gamma$ is

$$\Lambda_\gamma = \frac{\gamma \cdot C \cdot \overline{TC}_\gamma}{N \cdot l_M}, \tag{22}$$

and

$$\Lambda_\gamma = (\lambda_s + \lambda_{ua}) \cdot \overline{TC}_\gamma. \tag{23}$$

Then

$$\lambda_s + \lambda_{ua} = \frac{\gamma \cdot C}{N \cdot l_M}, \tag{24}$$

and

$$\overline{TC}_\gamma = N \cdot X_T + N \cdot X_M \cdot \Lambda_\gamma, \tag{25}$$

and

$$\overline{TC}_{\gamma} = N \cdot X_T + N \cdot X_M \cdot (\lambda_s + \lambda_{ua}) \cdot \overline{TC}_{\gamma} \tag{26}$$

Now solving for $eff\,(TRT_{ua}) = \overline{TC}_{\gamma}$,

$$eff\,(TRT_{ua}) = \frac{N \cdot X_T}{1 - N \cdot X_M \cdot (\lambda_s + \lambda_{ua})} \tag{27}$$

which is the effective value of the Urgent_Asynchronous *Target_Rotation_Time*.

As the offered load increases, increasing the network throughput beyond γ, the token cycle time increases beyond the TRT of the Urgent_Asynchronous access_class. Hence service to the Urgent_Asynchronous class gets reduced and finally drops to zero when TCT_{ua} equals TRT_{ua} as given by equation (2).

1.2. Results

In order to verify the analytic predictions, a 32 station network with each station offering service at all four access_classes, with identical traffic at all servers of an access_class, was simulated using the simulation package reported in [Summers 85] with parameters:

Bus capacity = 10 Mbps,

Size of data frames = 272 bits including framing,

Token = 112 bits,

X_T = 16.2 microseconds (including 50 bit times of propagation delay),

N = 32,

X_M = .0000272 seconds = 27.2 microseconds,

X_T = .0000162 seconds = 16.2 microseconds.

The individual message arrival rates were assumed such that they satisfied the conditions derived in the previous section. From the effective values of TRTs calculated using the equations derived in the previous section, the actual values of TRTs are given by $(eff\,(TRT_{asy}) - f\,(1) + 1\ octet\_time)$, where f(1) is equal to 34 octet_times, $eff\,(TRT_{ta})$ is 790 octet_times, $eff\,(TRT_{na})$ is 852 octet_times, $eff\,(TRT_{ua})$ is 1157 octet_times, and TRT_{ta} is 0.6056 msecs, TRT_{na} is 0.655 msecs, and TRT_{ua} is 0.8992 msecs.

The TRTs were set to the actual values shown above. A series of network configurations were simulated to show the variation of different parameters with throughput. The individual message arrival rates at each access_class have been varied from one simulation to another to increase the total offered load.

It can be observed from Figure 1 that average service time at the Time_Available access_class increases until a throughput of about 0.18. As the throughput increases further, the average service time starts falling and eventually drops to zero. From Figure 2 it can be observed that the average delivery times are reasonably low until the network throughput exceeds 0.18. This shows that most messages are getting transmitted on the same token cycle as they arrive. Beyond a throughput of 0.18 the token cycle time has exceeded TRT_{ta}. This can be observed from Figure 7. The Time_Available class is getting reduced service and messages are suffering higher queueing delays, so the delivery times are increasing exponentially in this region. This can be observed from Figure 2.

The TRT_{ta} value was calculated so as to achieve maximum possible service until a throughput of 0.18. The results of the simulation agree with the expected values very closely.

It can be observed from Figure 3 that the average service time at the Normal_Asynchronous access_class increases until a throughput of about 0.24. As the throughput increases further, the average service time starts decreasing and drops to zero. From Figure 4 it can be observed that the average delivery times are reasonably low until the network throughput is 0.24. This indicates that most messages are getting transmitted on the same token cycle as they arrive. Beyond a throughput of 0.24 the token cycle time has exceeded TRT_{na}. This can be seen from Figure 7. The Normal_Asynchronous class is getting reduced service leading to higher queueing delays, and the delivery times rise exponentially in this region. This can be observed from Figure 4.

The TRT_{na} value was calculated so as to achieve maximum possible service until a throughput of 0.24. From the above discussion it follows that the results of the simulation agree with the expected values very closely.

It can be observed from Figure 5 that the average service time at the Urgent_Asynchronous access_class increases until a throughput of about 0.44. As the throughput increases further, the average service time starts decreasing. From Figure 6 it can be observed that the average delivery times are reasonably low until the network throughput is 0.42. This indicates that most messages are getting transmitted on the same token cycle as they arrive. Beyond a throughput of 0.42 the token cycle time has exceeded TRT_{ua} and hence the Urgent_Asynchronous class is getting reduced service. This can be observed from Figure 7. Messages are suffering higher queueing delays and delivery times rise exponentially in this region, as seen in Figure 6.

The calculations showed that maximum possible service can be obtained at the Urgent_Asynchronous access_class until a throughput of 0.44. The same is illustrated by the simulation results.

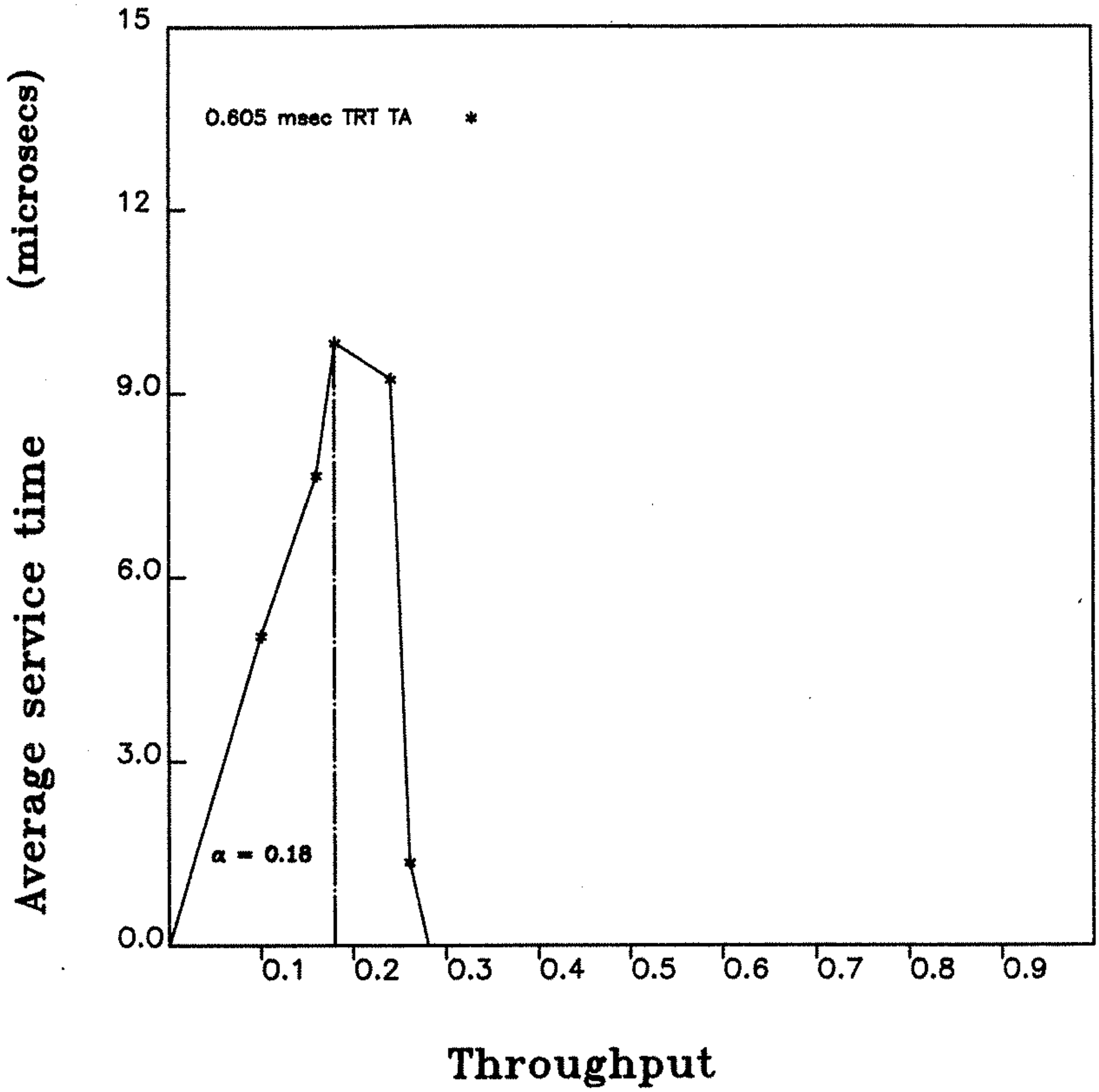

Figure 1
Average service time for Time_Available class

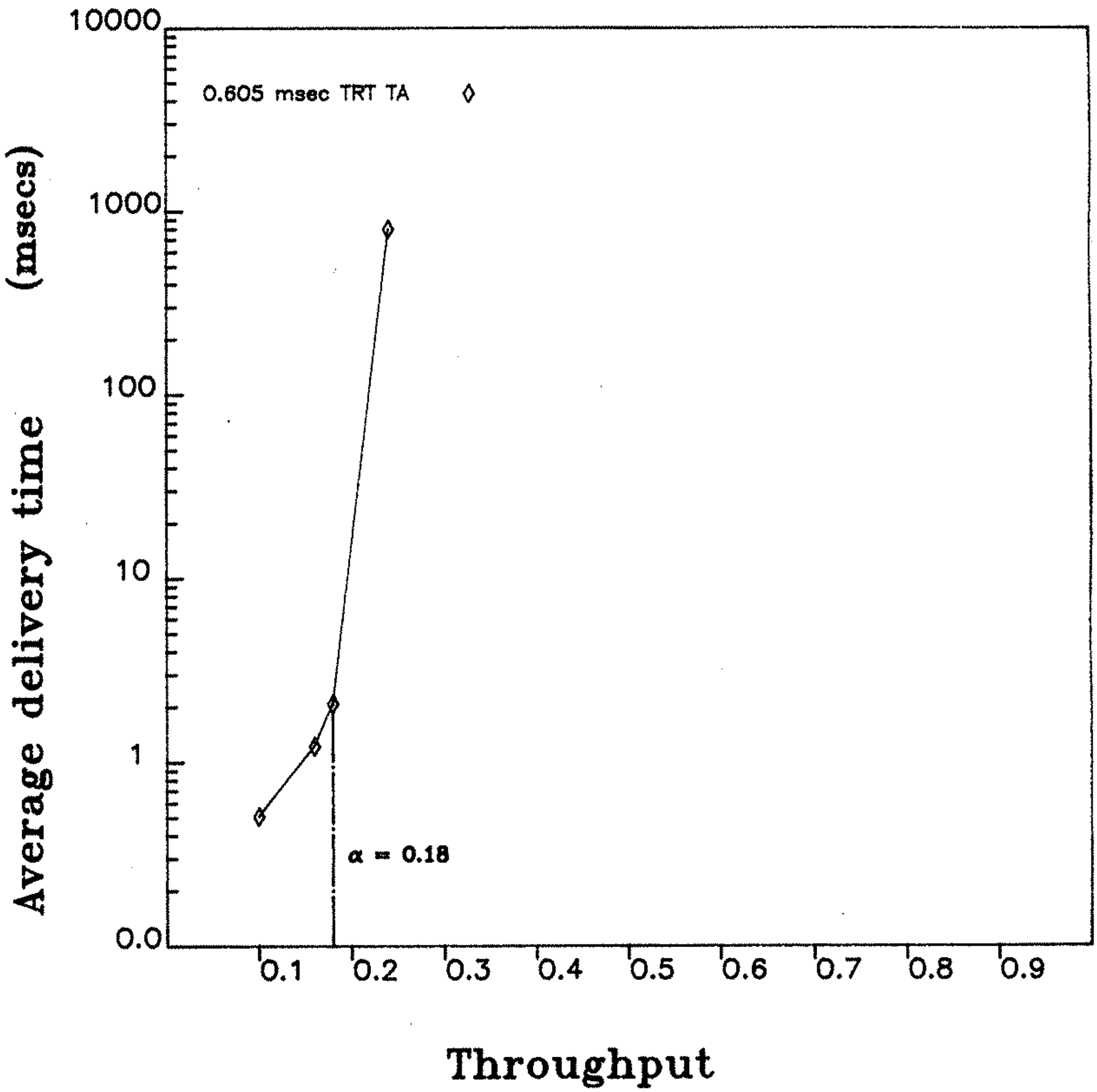

Figure 2
Average delivery time for Time_Available class

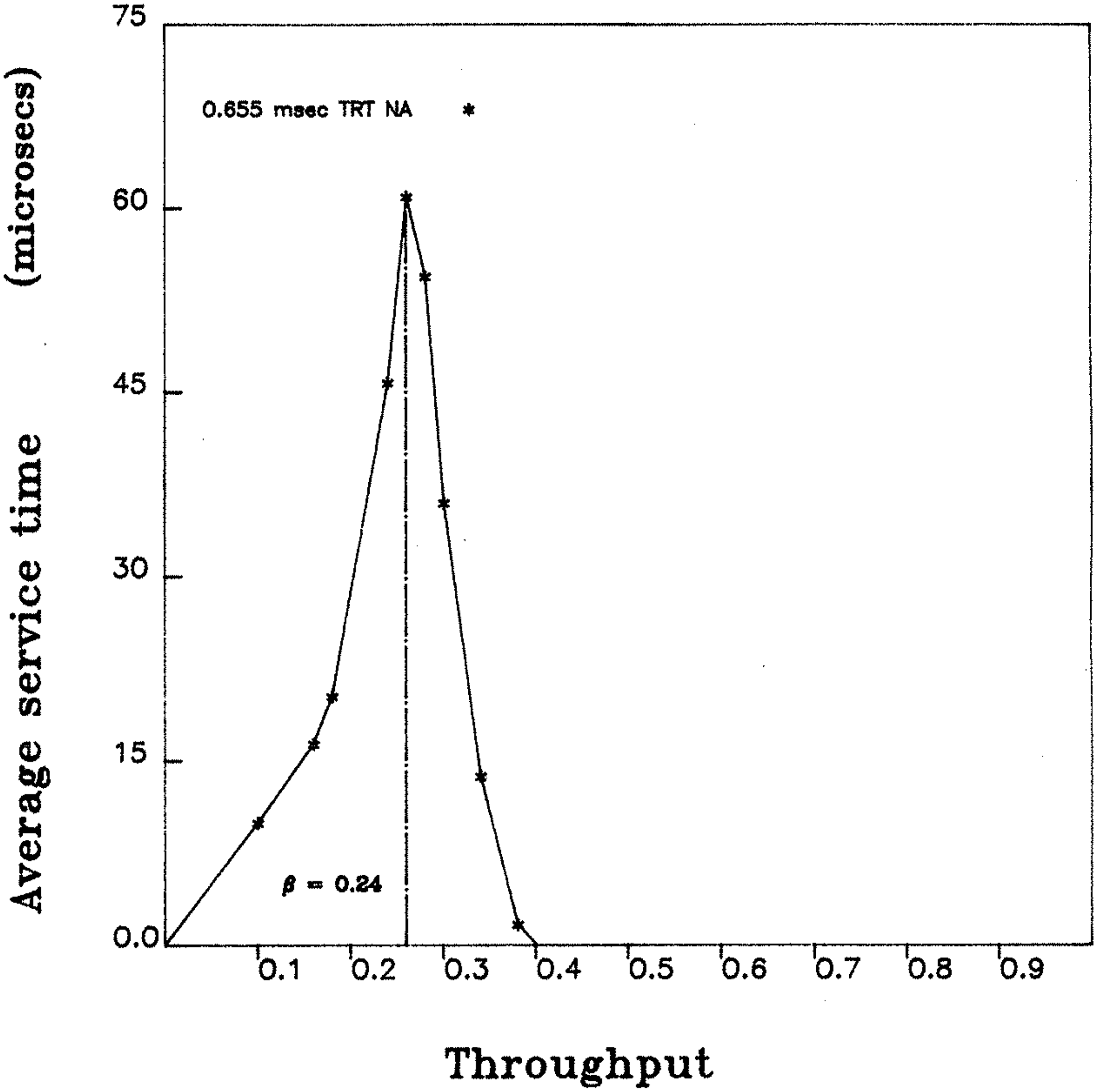

Figure 3
Average service time for Normal_Asynchronous class

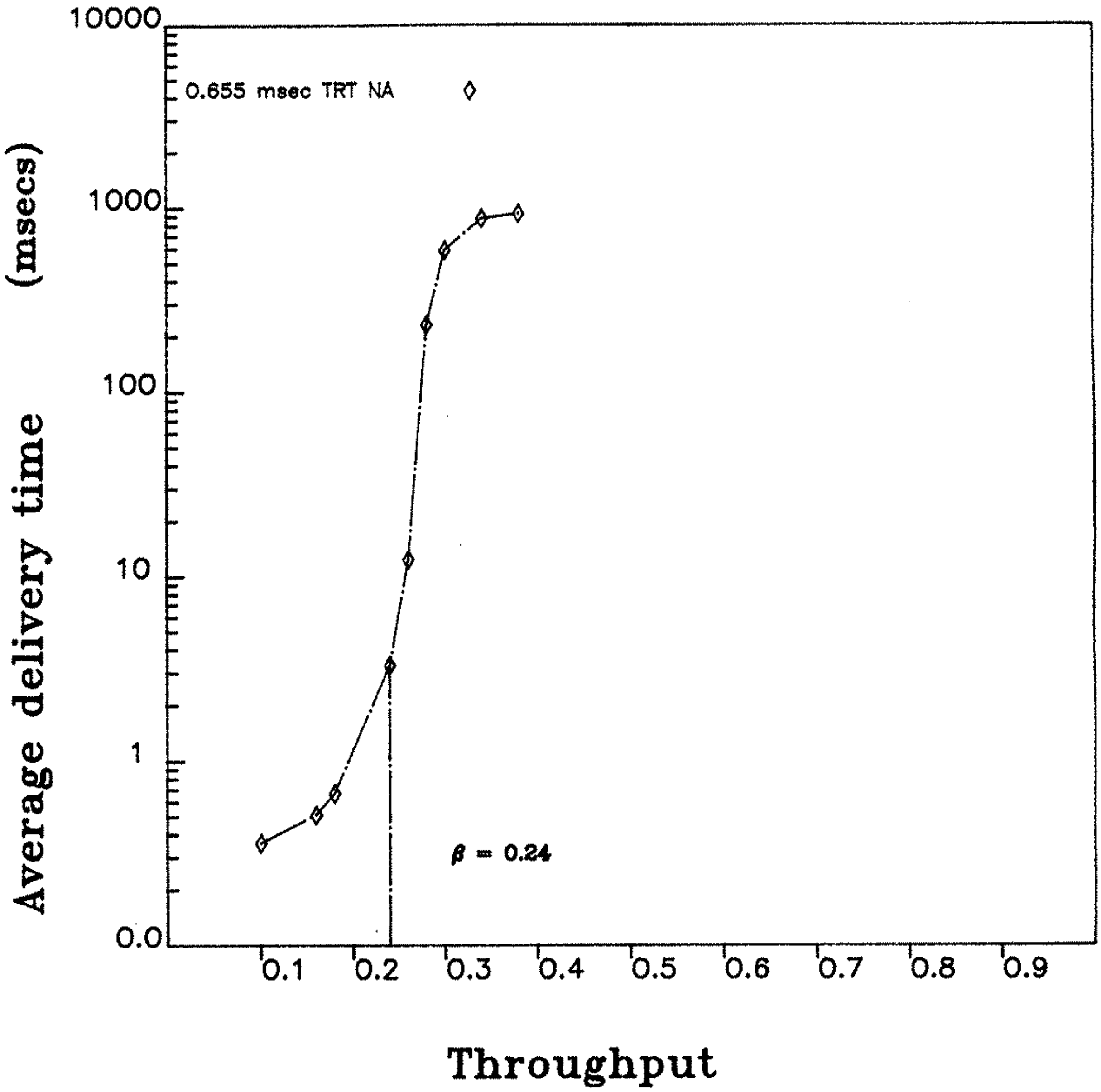

Figure 4
Average delivery time for Normal_Asynchronous class

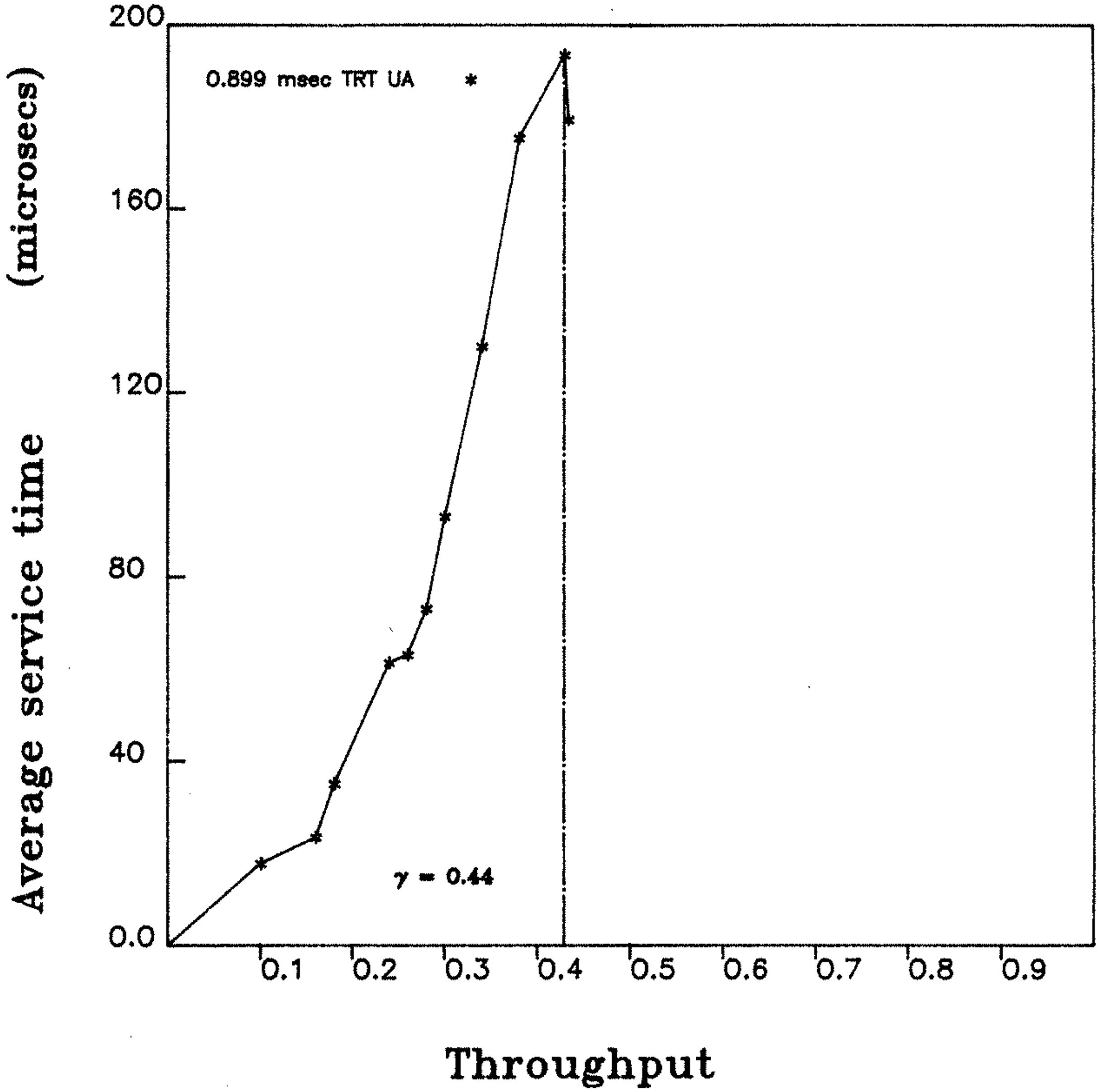

Figure 5
Average service time for Urgent_Asynchronous class

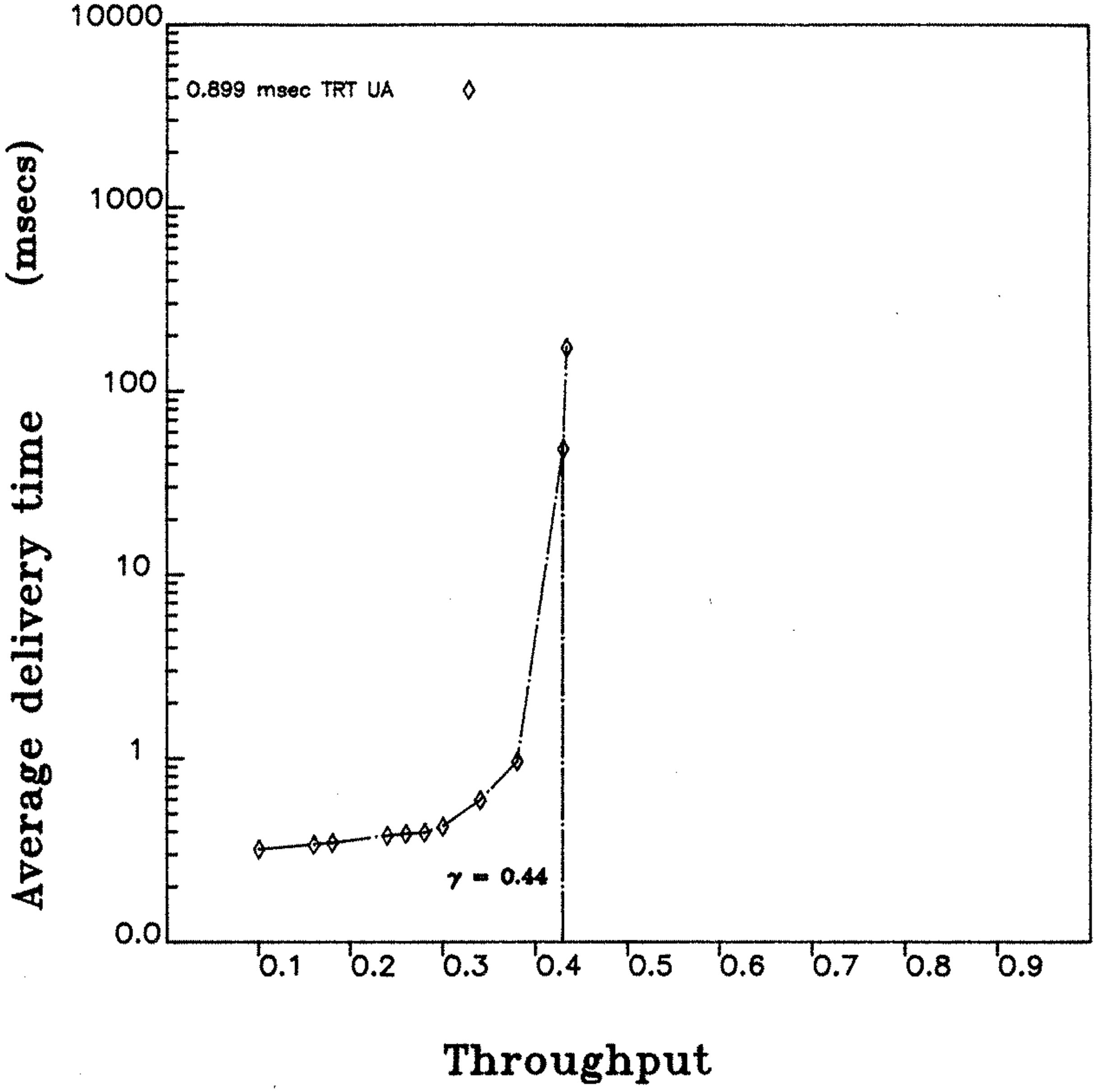

Figure 6
Average delivery time for Urgent_Asynchronous class

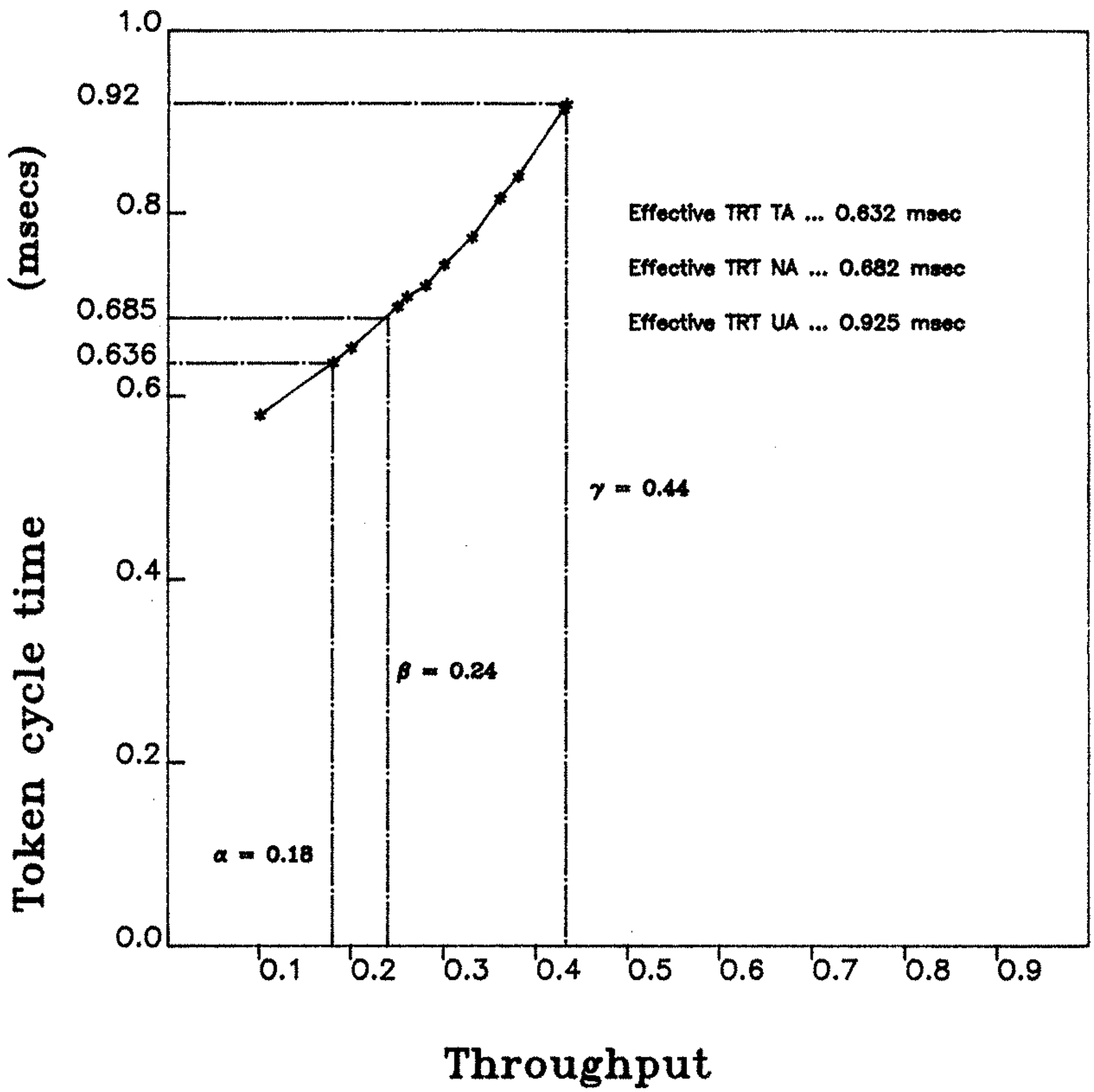

Figure 7
Token cycle time

1.3. Concluding Remarks

A static token passing bus has been studied and it has been shown that TRT settings play a crucial role in deciding the service time at the lower three access_classes. Optimum service can be obtained at an access_class until network throughput rises to the point that token cycle time equals the Target_Rotation_Time of that access_class. The analytic model developed in this study can be used to determine TRTs for any desired throughput range for a network configuration.

In this study all participating stations were assumed to be active and with identical message interarrival rates at all stations at an access_class. However the designer of a local area network has to take into consideration the communication traffic at each station at each access_class as it may be specific to each station and each access_class. Also the traffic from each station can be very bursty in nature. Nevertheless this study helped us gain a better understanding of the impact of certain parameters critical to the implementation of the priority feature.

Bibliography

[IEEE 1985]

Token-Passing Bus Access Method and Physical Layer Specifications, IEEE Standard 802.4-1985.

[Gorur 86]

Mangala Gorur and Alfred C. Weaver, *Priorities and Dynamic Ring Membership in an 802.4 Network*, Master's thesis, DAMACS Report No. 86-19, University of Virginia, Department of Computer Science, July 86.

[Summers 85]

Catherine F. Summers and Alfred C. Weaver, *Performance Studies of the IEEE Token Bus*, Master's thesis, DAMACS Report No. 85-06, University of Virginia, Department of Computer Science, May 1985.

[Weaver 85]

Catherine F. Summers and Alfred C. Weaver, *The IEEE 802.4 Token Bus - An Introduction and Performance Analysis*, Computer Science Report No. TR-85-19, University of Virginia, August 26, 1985.