OPTIMAL GUARANTEED SERVICES TIMED TOKEN (OGSTT) MEDIA ACCESS CONTROL (MAC) PROTOCOL FOR NETWORKS THAT SUPPORT HARD REAL-TIME AND NON REAL-TIME TRAFFIC

S. Ozuomba¹, C. O. Amaefule², J. J. Afolayan³

¹²³DEPT. OF ELECTRICAL/ELECTRONIC AND COMPUTER ENGINEERING, UNIVERSITY OF UYO, AKWA IBOM, NIGERIA
²DEPARTMENT OF ELECTRICAL ENGINEERING, IMO STATE UNIVERSITY, OWERRI, NIGERIA

E-mails addresses: ¹simeonoz@yahoo.com, ²littlekrix222@yahoo.com, ³afolayan.jimoh@yahoo.com

Abstract

In networks that support real-time traffic and non-real-time traffic over the same physical infrastructure, the challenge to the Media Access Control (MAC) protocol of such network is the ability to support the different traffic without compromising quality of service (QoS) for any of them. Generally, timed-token MAC protocols group the diverse real-time traffic into one category and then dedicate certain portion of the available bandwidth to them. At the same time, some bandwidth are left unassigned but available to the non real-time traffic. The unassigned bandwidth, and in some cases, the unused bandwidth left by the real-time traffic are assigned to the non-real-time traffic on best effort basis. In this paper, Optimal Guaranteed Services Timed Token (OGSTT) MAC protocol is developed and analyzed. In order to provide better support for both real-time traffic and non-real-time traffic on the same local area network, OGSTT employs the timed-token mechanisms in the Timely-Token protocol along with that of Budget Sharing Token (BuST) protocol. Some bounds on the behavior of OGSTT protocol are discussed along with the ability of OGSTT protocol to support real-time and non-real-time traffic. In particular, the paper demonstrated that the performance achieved by OGSTT is better than the Timely-Token and BuST. Furthermore, OGSTT protocol can be incorporated into the Ethernet network to provide real-time performance guarantees to multimedia applications and hard and soft real-time traffic.

Keywords: Timed-Token, Protocol, Ethernet, Timely-Token, multiservice, Networks, Real-Time, Non-Real-Time Traffic, Media, Access Control, quality of service.

1. Introduction

Nowadays, there is a rapid advent and advancements of many new and exciting applications: image processing and transmission, multimedia communications, office and factory automation, embedded real-time distributed systems, space vehicle systems, and the integration of expert systems into avionics and industrial process controls. The situation has placed an increasing demand for effective and efficient multi-services local area networks. Such networks’ MAC protocols must deal with different traffic patterns and must provide not only bounded message transmission time, as required by the hard real-time tasks, but also high throughput, as demanded by soft real-time and other non-real-time tasks [1]. An attractive approach for integrating such traffic is the timed-token protocol. Consequently, the timed-token protocol has been incorporated into several high-bandwidth network standards [2], including IEEE802.4 Token Bus [3], FDDI [4-8], SAFENET [9], Manufacturing Automation Protocol (MAP) [10], High-Speed Ring Bus [11] and in PROFIBUS which is a Fieldbus network standard [12]. The idea behind the Timed-Token protocol is, first, separate the messages generated in the system at run time into two classes, namely, real-time and non-real-time messages. Real-time messages are transmitted periodically and have a deadline, while non-real-time messages are transmitted on a best-effort basis. During initialization, the Target Token Rotation Time (TTRT) is selected. TTRT represents the expected time needed by the token to complete an entire round-trip of the network. Each node i is allocated $h_i$ time units (bandwidth) which is a portion of the TTRT; whenever a node receives the token, it can transmit its real-time messages for a time not greater than $h_i$. It can then transmit its non-real-time messages if the time elapsed since the previous token departure from the same node is less than the value of TTRT, that is, only if the token arrives earlier than expected. FDDI timed-token is one of the earliest timed-token passing protocol. In FDDI, the token rotation time may reach 2(TTRT) [6]. Due to this token lateness,
an FDDI network can use at most half of its bandwidth to transmit real-time traffic [5, 13, 14]. To alleviate this deficiency, Shin et al. proposed the FDDI-M token protocol [5]. In FDDI-M, the token is never late. This allows FDDI-M to double FDDI's ability to support real-time traffic. However, FDDI-M has one major weakness; starvation of non-real-time traffic. This means that in some cases, FDDI-M may not be able to transmit non real-time traffic. Budget Sharing Token (BuST) protocol [14, 15] and Timely-Token [13] protocols are timed-token protocols recently introduced to improve the communication services provided by FDDI and FDDI-M networks. The BuST and Timely-Token solved the problems of token-lateness in FDDI and the starvation of non-real-time traffic in FDDI-M.

The network and message models are presented in Section 2. Timely-Token and BuST protocols, along with their weaknesses, are described in Section 3. The OGSTT protocol is described in Section 4. Also, the performance bounds of the OGSTT protocol is presented in section 4. Section 5 compares the OGSTT against Timely-Token and BuST protocols. Also, in section 5, sample numerical example and discussion of results are presented. Finally, conclusion and recommendations for further studies are given in Section 6.

2. Review of Relevant Literature

2.1 Network Model

The timed-token protocols in this paper operate on a token ring network consisting of N nodes. Each node has a unique number in the range 0, 1, 2...N-1. For each node i, the next node is node (i+1) or more appropriately node (i+1) mod N. The token frame circulates around the ring from node i to node i + 1, i + 2, ... until node i + (N-1), then to nodes i, i + 1, i+2, ..., etc. Let \( w_i \) denote the latency or walk-time between a node i and its upstream neighbor node (i + 1). The sum of all such latencies in the ring is known as the ring latency or the token walk-time, \( W \), where \( W = \sum w_i \).

2.2 Message Model

Messages generated in the system at run time may be classified as either real-time messages or non-real-time messages. Agrawal et al. [16] showed how a token-ring network having multiple real-time streams per station could be transformed into a logically-equivalent network with one real-time message stream per station. Therefore, without loss of generality, a single real-time message stream per station is assumed. The real-time message stream of station i is denoted by the triple \( (P_i, D_i, C_i) \). Message length, \( C_i \), is the amount of time needed to transmit a maximum size message. Period length, \( P_i \), is the minimum inter-arrival period for the real-time message stream at node i. Message deadline; \( D_i \), is the maximum amount of time that can elapse between a message arrival and the completion of its transmission. Thus, if a message stream arrives at time \( t \), then it must be transmitted by time \( t + D_i \).

2.3 Operation of the Existing Timed-Token Protocols

Generally, in the timed-token protocols, during the initialization, each node i declares a Target Token Rotation Time, TTRT. The minimum declared value is selected as the ring's TTRT. Each node i is then assigned a portion \( h_i \) of the TTRT to transmit its real-time traffic. When a node receives the token, it can transmit its real-time traffic for a time not greater than \( h_i \) time units. However, to initialize the timers, no packets are transmitted during the first token rotation. The main difference among the various timed-token protocols concerns the non-real-time message service. Let \( H \) be defined as, \( \sum h_i \), where \( \sum h_i \) is the sum of the time reserved for the real-time traffic in all the nodes in every cycle. Let \( T = H + W \), where \( T \) is the total time allocated per cycle to the real-time traffic and walk-time. The value of TTRT is denoted as \( \tau \). In the timed-token protocols, there are two categories of bandwidths that can be used by the non-real-time traffic, namely;

**Category I:** (\( \tau - T \)) which is the total bandwidth that is not allocated to the real-time traffic and ring latency.

**\( \tau - T \)** bandwidth (time units) is available to the non-real-time traffic in every cycle. Let \( A^* = \tau - T \).

**Category II:** (U) which is the bandwidth that is allocated to the real-time traffic but not used by the real-time traffic in the previous cycle.

The different timed-token protocols differ in the way they allocate the two categories of available bandwidth to the non-real-time traffic. As shown in [13-15], [17], [18], the timely-token protocol [13] and the BuST protocol [15] improved on the ability of the FDDI and FDDI-M timed-token protocols to support real-time and non-real-time traffic.

2.4 Non real-time Traffic Transmission Mechanism in the Timely-Token Protocol

In FDDI and FDDI-M protocols, problems occurred because a station cannot distinguish between unused real-time bandwidth and unused non-real-
Drawbacks of Timely-Token Protocol

In the Timely-Token, non-real-time traffic makes use of only Category I available bandwidth. The Timely-Token does not permit the non-real-time traffic to use the spare bandwidth (i.e., U) left over by the real-time traffic. As such, the throughput of the Timely-Token decreases when U > 0.

2.5 Non real-time Traffic Transmission Mechanism in The BuST Protocol

In the BuST, a node can deliver non-real-time traffic each time it gets the token, early or not, using the spare bandwidth (i.e., U) left by the real-time traffic. If \( s_i \) is the time units consumed by node i to deliver real-time traffic, then it can send non-real-time traffic for a time not greater than \( h_i - s_i \) time units even if the token is not early.

Drawbacks of BuST Protocol

In BuST protocol, the non-real-time traffic makes use of only Category II available bandwidth. As such, when the load level of the real-time traffic is heavy, \( s_i = h_i \), then, no bandwidth will be left for the non-real-time traffic. In that case, non-real-time traffic will be starved. Besides, Category II bandwidth is not allocated in such a way that the unused bandwidth in a node can be used by the non-real-time traffic in another node. So, while some nodes with light load of real-time and non-real-time traffic may have spare bandwidth left over, the other nodes with heavy load of real-time traffic will still starve their non-real-time traffic as they cannot use the spare bandwidth from other nodes.

3. Methodology

In this paper, Optimal Guaranteed Services Timed Token (OGSTT) MAC protocol was proposed for networks that support hard real-time and non-real-time traffic. The analytical modeling approach is adopted to develop and evaluate the proposed MAC protocol. Mathematical expressions are developed for various performance parameters for the proposed MAC protocol. Then, the performance of the MAC protocol is evaluated with MatLab software. Specifically, with the help of the MatLab software, the mathematical expressions derived for each of the performance parameters is used to compute its value under various network and traffic configurations. The parameters for the performance analysis include; Average Bandwidth Used by the Real-time Traffic Per Cycle, Average Bandwidth Used by the Non Real-time Traffic Per Cycle, and Average Cycle Length. These performance parameters are then used to compare the performance of the proposed MAC protocol with those of some existing timed-token passing MAC protocols. The comparison is based on the ability of each of the MAC protocol to support the non-real-time traffic for any given load level of the real-time traffic.

4. Outline of the OGSTT Protocol

(a) During the ring initialization phase, each node i declares a TTRT. The minimum declared value is selected as the ring's TTRT. Each node i is then assigned a portion \( h_i \) of the TTRT to transmit its real-time traffic in every cycle. During each token rotation, station i can transmit real-time packets for at most \( h_i \) time units.

(b) Each station i has a token-rotation timer, TRT, for measuring the time between token arrivals.

(c) Each station i has a non-real-time -limit variable, \( A_i \). In this variable, station i stores the amount of time it may transmit non-real-time messages. In addition, station i maintains a variable \( A_i \), where it stores the portion of \( h_i \) the reserved real-time bandwidth it used in transmitting real-time traffic in the previous token-rotation. Also, another variable, \( b_i \), is defined, where station i stores the portion of \( h_i \) the reserved real-time bandwidth station i used in transmitting non-real-time traffic in the previous token rotation. Also, station i maintains a variable \( s_i \), where it stores the total time units used out of \( h_i \) in the previous token-rotation, where \( s_i = A_i + b_i \).

(d) To initialize the token-rotation timers, no packets are transmitted during the first token rotation. In addition, \( s_i \) is set to zero for all i, and \( U = \sum_i h_i = H \). The integer U is added to the token, where U represents the sum of unused real-time bandwidth of all stations during the previous token-rotation. When the token arrives at station i, U should also include the unused real-time bandwidth of station i in the previous token-rotation.
When station \( i \) receives the token, it performs the following steps:

1. \( A_i := (TTRT - U - TRT_i)^+ \) (1)
2. \( \text{TRT}_i := 0 \) (2)
3. \( U := U - (h_i - s_i) \) (3)
4. If node \( i \) has real-time packets, it transmits them until \( \text{TRT}_i \) counts up to \( h_i \), or until all the real-time traffic is sent, whichever comes first.
5. \( \varphi_i \) is assigned the number of time units of real-time transmission used in step 4.
6. If \( \text{TRT}_i < h_i \), then if node \( i \) has a real-time packets, it transmits them until \( \text{TRT}_i \) counts up to \( h_i \), or until all the non-real-time traffic is sent, whichever comes first.
7. \( b_i \) is assigned the number of time units of non-real-time transmission used in step 6.
8. \( s_i \) is assigned the total number of time units of real-time and non-real-time transmissions used in step 4 and step 6.
9. \( U := U + (h_i - s_i) \)
10. If station \( i \) has non real-time packets, it transmits them for a time period of up to \( A_i \) time units, or until all its non-real-time packets are transmitted, whichever occurs first.
11. Station \( i \) passes the token to station \( (i + 1) \mod N \).

### 4.1 Performance Bounds OGSTT Protocol

In principle, OGSTT operates like a heavily loaded Timely-Token protocol. The difference lies in how OGSTT and the Timely-Token handle \( U \), the drop in load of real-time traffic. In OGSTT protocol, Category I (i.e. \( A^* \)) available bandwidths are allocated to the non-real-time traffic just like in the Timely-Token protocol. At the same time, Category II \( U \) spare bandwidths left over by the real-time traffic are allocated to the non-real-time traffic just like in the BuST protocol. Consequently, maximum throughput is maintained by OGSTT even in the face of drop or variation in the load level of the real-time traffic.

Technically, the difference between OGSTT and Timely-Token is that in the Timely-Token \( s_i = \varphi_i \) whereas in the OGSTT \( s_i = \varphi_i + b_i \). As such, analysis of the OGSTT protocol is simply the analysis of the heavily loaded Timely-Token system where \( s_i \) is composed of \( \varphi_i \) and \( b_i \), the bandwidths used by the real-time and non-real-time traffic respectively. Hence, in the analysis, the approach employed for the heavily loaded Timely-Token in [13] is adopted. There is however one slight difference in the assumption made here. In [13], the system is assumed to be heavily loaded with real-time and non-real-time traffic. In this paper, the system is loaded with light load of real-time traffic but with heavy load of non-real-time traffic. As such, in this paper, in every token receipt, the real-time traffic may not use all the time units reserved for it in the node. However, the unused portions of the reserved real-time time units are used by the non-real-time traffic in every node. In this way, the system still behaves like a heavily loaded system since all the time units for data transmission are used up in every node in every token receipt.

In order to reason about values that change over time, the notations used for the analysis are enhanced to include the following terms presented below.

### 4.2 Definition of terms

\( R^{lm} \): round \( m \) of station \( i \), i.e., time interval \([t, t']\), where \( t \) is the time when station \( i \) receives the token for the \( m \)th time, and \( t' \) is the time when station \( i \) receives the token for the \((m + 1)\)th time.

\( A_i^{lm} \) : value assigned to \( A_i \) during \( R^{lm} \). In particular, \( A_i^{1m} \) is the value assigned to \( A_i \) when the token is received at the beginning of \( R^{lm} \).

\( d_j^{lm} \) : duration of non-real-time transmission of station \( j \) during \( R^{lm} \). Note that \( d_j^{lm} \leq A_j^{lm} \) [13].

\( h_j \) : duration of time units reserved for real-time transmission of station \( j \) in every round.

\( \varphi_j^{lm} \) : the portion of the \( h_j \) time units actually used for real-time transmission in station \( j \) during \( R^{lm} \). Note that [13]:

\[
\varphi_j^{lm} \leq s_j^{lm} \leq h_j \quad (4a)
\]

\( b_j^{lm} \) : the portion of the \( h_j \) time units actually used for non-real-time transmission in station \( j \) during \( R^{lm} \). Note that:

\[
b_j^{lm} \leq s_j^{lm} \leq h_j \quad (4b)
\]

\( s_j^{lm} \) : the total of the portions of the \( h_j \) time units actually used for real-time and non-real-time transmissions in station \( j \) during \( R^{lm} \). Note that \( s_j^{lm} \leq h_j \) [13]. Also,

\[
s_j^{lm} = \varphi_j^{lm} + b_j^{lm} \leq h_j \quad (5)
\]

\( TRT_j^{lm} \) : value of \( \text{TRT}_j \) when station \( j \) receives the token during \( R^{lm} \). In particular, \( TRT_j^{lm} \) is the value of \( \text{TRT}_j \) when the token is received at the beginning of \( R^{lm} \) [13].

\[
TRT_j^{lm} = s_j^{lm-1} + d_j^{lm-1} + W \quad (6)
\]

### 4.3 Theorem 1 (The Token is never late)

For every station \( i \), upon token arrival, \( \text{TRT}_i \leq TTRT \).

The proof for Theorem 1 is given in [13]. The same applies to OGSTT protocol. It was shown in [13] that...
for the heavily loaded Timely-Token protocol, the following expressions hold:
\[ \sum_{i} a_i^{m-1} \leq A \]
\[ U = \sum_{i} h_i - \sum_{i} s_i^{m-1} \text{ and } \sum_{i} s_i^{m-1} \leq h_i \]
From the discussion in this paper, it can be seen that for the Timely-Token protocol [13],
\[ \sum_{i} s_i^{m-1} = \sum_{i} \varphi_i^{m-1} \]
whereas, for the OGSTT protocol,
\[ \sum_{i} s_i^{m-1} = \sum_{i} \varphi_i^{m-1} + \sum_{i} b_i^{m-1} \leq \sum_{i} h_i \]
Then,
\[ TRT_i^{m} = s_i^{m-1} + \varphi_i^{m-1} + W \]
\[ TRT_i^{m} = \sum_{i} \varphi_i^{m-1} + \sum_{i} b_i^{m-1} + a_i^{m-1} + W \]
\[ TRT_i^{m} \leq \sum_{i} h_i + A^* + W \leq TTTRT \]

So, the token is never late since the Token Rotation Time, TRT, does not exceed TTTRT.

5. Comparison of OGSTT against the Timely-Token and BuST Protocols

In this section, the BuST protocol is compared against the Timely-Token and BuST Protocols. The comparison focuses on the ability of these protocols to support real-time and non-real-time traffic. The comparison is based on the expression for the upper bound on the average cycle length (\( \bar{C} \)) for these protocols, because the expressions directly reflect the ability of the protocol to provide services to the real-time and non-real-time traffic.

5.1. Expression for the Upper Bound on the Average Cycle Length (\( \bar{C} \))

5.1.1 FDDI Protocol

In FDDI timed-token protocol [6, 7, 19, 20], each node has two timers, the token holding timer (THT) and the token-rotation-timer (TRT). The TRT counter always increases, whereas the THT only increases when the node is delivering non-real-time traffic. When TRT reaches TTTRT, it is reset to 0 and the token is considered as late by incrementing the node’s late count, Lci by one. The actual token cycle time, denoted in this paper as TRT* is given as;
\[ TRT_i^* = TTTRT_i + Lci \text{ (TTTRT)} \]
The token is considered to arrive early at node i if Lci = 0 otherwise the token is late (in this case, Lci ≥ 1). When the token arrives at a node, the node can transmit non-real-time traffic for a time no greater than THT; where THT is given as;
\[ THT_i = TTTRT_i - TRT_i^* \]
\[ \text{for } TRT_i^* < TTTRT \text{ otherwise } THT_i = 0; \]
where TRT* is the time spent in the last round-trip of the token. Then, for the FDDI, \( A_i = \max(0, TTTRT_i \cdot \varphi_i) \).

Joseph and Fouad has shown in [19] that for FDDI protocol, the upper bound on the average cycle length (\( \bar{C} \)) for a heavily load system is given as
\[ \bar{C} \leq \left[ \frac{N}{N+1} \right] (\tau - T) + T \]
Then, the upper bound on the average bandwidth allocated to the non-real-time traffic (\( \bar{A} \)) is given as
\[ \bar{A} = \left[ \frac{N}{N+1} \right] (\tau - T) \]
Similarly, Ozuomba and Chukwudebe showed in [20] that for FDDI protocol, \( \bar{C} \) and \( \bar{A} \) for a system with light load of real-time traffic but with heavy load of non-real-time traffic, are defined as follows;
\[ \bar{C} \leq \left[ \frac{N}{N+1} \right] (\tau - T) + \left[ \frac{N}{N+1} \right] U + (H - U) + W \]
where U is the unused real-time transmission time in the last round-trip of the token. The assumption made in [20] is that U is constant for at least the N+1 consecutive cycle where the average is taken.

5.1.1 Timely-Token Protocol

The difference between the FDDI and the Timely-Token is in the use of TTTRT in the FDDI and TTTRT* in the Timely-Token protocol, where TTTRT* = TTTRT - U. For the Timely-Token, \( U = \sum_i (h_i - s_i) \) and \( s_i = \varphi_i \)
then, \( \tau \) can be replaced with \( \tau \cdot U \) in the expressions for \( \bar{C} \) in Eq 17 and \( \bar{A} \) in Eq 18 to obtain \( \bar{C}_T \) and \( \bar{A}_T \) for the Timely-Token, where \( \sum_i \varphi_i = \sum_{i=0}^{N-1} \varphi_i \) and \( \lim_{i \to \infty} \left[ \frac{N}{N+1} \right] = \frac{N}{N+1} \) then,
\[ \bar{C}_T \leq \left[ \frac{N}{N+1} \right] (\tau - T) + \sum_i \varphi_i + W \]
\[ \bar{A}_T = \left[ \frac{N}{N+1} \right] (\tau - T) \]

5.1.3 BuST Protocol

In the BuST protocol, Category I available bandwidth (i.e. (\( \tau \cdot T \)) is not used by any traffic. The non-real-time traffic makes use of only the Category II which is the U spare bandwidth left over by the real-time traffic. So, \( \text{THT}_i = 0 \) for i and \( A_i = U \). Now \( U = \sum_i (h_i - s_i) \) and \( s_i = \varphi_i \), thus, \( \bar{C}_B \) and \( \bar{A}_B \) for the BuST protocol are given as follows;
\[ \bar{C}_B \leq U + \sum_i \varphi_i + W \]
\[ \bar{A}_B = U = H - \sum_i \varphi_i \]
5.1.4 OGSTT Protocol
For the OGSTT protocol, TTRT* = TTRT, then
\[ U = \sum_i (h_i + S_i) \] (23a)
and
\[ S_i = \varphi_i + b_i \] (23b).
Since a system with heavy load of non-real-time traffic is being considered, then,
\[ H = \sum_i b_i + \sum_i \varphi_i \] (24)
can be replaced with \( r \cdot U \) in the expressions for \( \hat{C} \) in Eq 17 and \( \hat{A} \) in Eq 18 to obtain \( \hat{C}_G \) and \( \hat{A}_G \) as follows;
\[ \hat{C}_G \leq \left[ \frac{N}{N+1} \right] - T + \sum_i b_i + \sum_i \varphi_i + w \] (25)
\[ \hat{A}_G \leq \left[ \frac{N}{N+1} \right] (\tau - T) + (H - \sum_i \varphi_i) \] (26)
\[ \hat{C}_G \leq \left[ \frac{N}{N+1} \right] (\tau - T) + \sum_i b_i \] (27a)
\[ \hat{A}_G \leq \left[ \frac{N}{N+1} \right] (\tau - T) + (H - \sum_i \varphi_i) \] (27b)

5.2 Simulation Results
Consider a ring network with four nodes (i.e. \( N = 4 \)) where \( r = 100 \), \( W = 4 \) and \( h_i = 20 \) for all the nodes. It will be assumed that the network is heavily loaded with non-real-time traffic but with a variable load of the real-time traffic. The real-time traffic load, \( \varphi_i \) can vary from 0 to \( h_i \). The values of \( \hat{C} \) and \( \hat{A} \) for the various load levels of the real-time traffic are computed for the Timely-Token, BuST and OGSTT protocols. The results are presented in Table 1 and Table 2.

5.3 Discussion of results
5.3.1 A System With Heavy Load Of Real-time and Non-real-time Traffic
When there is heavy load of real-time traffic, that is \( \sum_i \varphi_i = H = 80 \), \( U = H - \sum_i \varphi_i = 0 \) then
(a) the BuST will not allocate bandwidth to the non-real-time traffic, that is \( \hat{A} = 0 \) (Table 1) and \( \hat{C} = 84 \) (Table 2).
(b) the Timely-Token will allocate a constant average bandwidth \( \left( \frac{N}{N+1} \right) A^* \) to the non-real-time traffic, where \( A^* = 16 \), \( N = 4 \), so \( \frac{N}{N+1} (A^*) = 12.8 \). So \( \hat{A} = 12.8 \) (Table 1) and \( \hat{C} = 96.8 \) (Table 2).
(c) the OGSTT will allocate an average bandwidth \( \left( H - \sum_i \varphi_i \right) + \frac{N}{N+1} A^* \) to the non-real-time traffic, where \( H - \sum_i \varphi_i = 0 \), \( A^* = 16 \), \( N = 4 \), so \( \frac{N}{N+1} (A^*) = 12.8 \). So, \( \hat{A} = 12.8 \) (Table 1) and \( \hat{C} = 96.8 \) (Table 2).

So, in the case of a system with heavy load of real-time and non-real-time traffic, the Timely-Token and the OGSTT have the same throughput which is higher than the BuST throughput. In particular, BuST will not allocate any bandwidth to the non-real-time traffic, in this case (\( \hat{A} = 0 \)).

<table>
<thead>
<tr>
<th>( \varphi_i )</th>
<th>( \Sigma \varphi_i )</th>
<th>( \hat{A} )</th>
<th>( \hat{C} )</th>
<th>( \hat{C}_T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>12.8</td>
<td>92.8</td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>0.8</td>
<td>12.8</td>
<td>84.8</td>
<td></td>
</tr>
<tr>
<td>0.4</td>
<td>1.6</td>
<td>12.8</td>
<td>76.8</td>
<td></td>
</tr>
<tr>
<td>0.6</td>
<td>2.4</td>
<td>12.8</td>
<td>68.8</td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>3.2</td>
<td>12.8</td>
<td>60.8</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>4.0</td>
<td>12.8</td>
<td>52.8</td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>4.8</td>
<td>12.8</td>
<td>44.8</td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>5.6</td>
<td>12.8</td>
<td>36.8</td>
<td></td>
</tr>
<tr>
<td>1.6</td>
<td>6.4</td>
<td>12.8</td>
<td>28.8</td>
<td></td>
</tr>
<tr>
<td>1.8</td>
<td>7.2</td>
<td>12.8</td>
<td>20.8</td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>8.0</td>
<td>12.8</td>
<td>12.8</td>
<td></td>
</tr>
</tbody>
</table>

Table 2 Comparison of the computed values of Average Cycle length, \( \hat{C} \) for BuST, Timely-Token and OGSTT.

<table>
<thead>
<tr>
<th>( \varphi_i )</th>
<th>( \Sigma \varphi_i )</th>
<th>( \hat{C} )</th>
<th>( \hat{C}_T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>12.8</td>
<td>96.8</td>
</tr>
<tr>
<td>0.2</td>
<td>0.8</td>
<td>24.8</td>
<td>96.8</td>
</tr>
<tr>
<td>0.4</td>
<td>1.6</td>
<td>32.8</td>
<td>96.8</td>
</tr>
<tr>
<td>0.6</td>
<td>2.4</td>
<td>40.8</td>
<td>96.8</td>
</tr>
<tr>
<td>0.8</td>
<td>3.2</td>
<td>48.8</td>
<td>96.8</td>
</tr>
<tr>
<td>1.0</td>
<td>4.0</td>
<td>56.8</td>
<td>96.8</td>
</tr>
<tr>
<td>1.2</td>
<td>4.8</td>
<td>64.8</td>
<td>96.8</td>
</tr>
<tr>
<td>1.4</td>
<td>5.6</td>
<td>72.8</td>
<td>96.8</td>
</tr>
<tr>
<td>1.6</td>
<td>6.4</td>
<td>80.8</td>
<td>96.8</td>
</tr>
<tr>
<td>1.8</td>
<td>7.2</td>
<td>88.8</td>
<td>96.8</td>
</tr>
<tr>
<td>2.0</td>
<td>8.0</td>
<td>96.8</td>
<td>96.8</td>
</tr>
</tbody>
</table>

5.3.2 A System With No Real-time Traffic But With Heavy Load Of Non-real-time Traffic
When there is no real-time traffic, that is \( \sum_i \varphi_i = 0 \);
\( H = 80 \), \( U = 80 \)
OGSTT Network Mac Protocol for Hard Real-time and Non-Real-time Traffic, S. Ozuomba et al

(a) the BuST will allocate all the (U) spare bandwidths to the non-real-time traffic, that is \( \bar{A} = 80 \) (Table 1) and \( \bar{C} = 84 \) (Table 2).

(b) the Timely-Token will allocate the same constant average bandwidth \( \frac{N}{N+1} (A^*) \) to the non-real-time traffic, where \( A^* = 16 \), \( N = 4 \), so \( \frac{N}{N+1} (A^*) = 12.8 \). So \( \bar{A} = 12.8 \) (Table 1) and \( \bar{C} = 16.8 \) (Table 2).

(c) the OGSTT will allocate an average bandwidth \( \left( \frac{H - \sum_i \varphi_i}{N} \frac{N}{N+1} (A^*) \right) \) to the non-real-time traffic, where \( H - \sum_i \varphi_i = 80 \), \( A^* = 16 \), \( N = 4 \), so \( \frac{N}{N+1} (A^*) = 12.8 \). Then, \( \bar{A} = 92.8 \) (Table 1) and \( \bar{C} = 96.8 \) (Table 2).

So, in the case of a system with no real-time traffic but with heavy load of non-real-time traffic, the Timely-Token will allocate the least amount of bandwidth to the non-real-time traffic while the OGSTT will allocate the highest. The BuST will allocate all the spare bandwidth \( (U = H \cdot \sum_i \varphi_i = 80) \) left unused by the real-time traffic to the non-real-time traffic \( (\bar{A} = H = 80) \).

5.3.3 A System with Variable Load Level of Real-time Traffic but with Heavy Load of Non-real-time Traffic

Generally, if there is heavy load of non-real-time traffic, then, as the load of the real-time traffic increases from zero (no real-time traffic) to the heavy load state, then,

(a) the Timely-Token allocates the same amount of average bandwidth \( \frac{N}{N+1} (A^*) \) to the non-real-time traffic. The overall throughput of the system increases but it is less than the achievable maximum value of \( H + \frac{N}{N+1} (A^*) \).

(b) the BuST allocates all the spare bandwidths \( (U = H - \sum_i \varphi_i) \) left over by real-time traffic to the non-real-time traffic. The overall throughput of the system remains the same as \( H \) but it is less than the achievable maximum value of \( H + \frac{N}{N+1} (A^*) \).

(c) the OGSTT allocates \( \frac{N}{N+1} (A^*) \) plus the spare bandwidth \( (U = H - \sum_i \varphi_i) \) left over by real-time traffic to the non-real-time traffic. The overall throughput of the system remains the same as the achievable maximum value of \( H + \frac{N}{N+1} (A^*) \).

So, in the case of a system with heavy load of non-real-time traffic but with variable load of real-time traffic, the OGSTT protocol maintains higher throughput than the Timely-Token and BuST protocols as long as \( \sum_i \varphi_i < H \).

6. Conclusion and Recommendations

6.1 Conclusion

This paper presented the Guaranteed Services Token protocol (OGSTT) which improved the performance of existing timed-token protocols, including the Timely-Token and BuST protocols. BuST and Timely-Token protocols are time-token protocols recently introduced to improve the communication services provided by FDDI and FDDI-M networks. However, OGSTT maintained higher throughput than BuST and Timely-Token protocols in the face of variations in the load level of the real-time traffic. At the same time, OGSTT delivered guaranteed services as required by the hard and soft real-time applications. Consequently, OGSTT is more suitable for multi-services networks since it can efficiently support different traffic patterns and also provide not only bounded message transmission time as required by the hard real-time tasks, but also high throughput, as demanded by soft real-time and non-real-time tasks.

6.2 Recommendations

OGSTT can be incorporated into Ethernet and Profinet networks to improve the performance of those networks. The approach to be adopted and the implementation issues are areas of further research.

References