We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

12

Token Passing Techniques for Hard Real-Time Communication

Gianluca Franchino*, Giorgio C. Buttazzo* and Tullio Facchinetti**.
*Scuola Superiore Sant'Anna, Italy
**University of Pavia, Italy

1. Introduction

Distributed computing platforms are increasingly used to develop critical embedded systems, like control applications, sensors networks, telecommunication, and robotics systems. In such distributed applications, the correct behaviour depends on the timely execution of the tasks running on different nodes, which may frequently exchange shared data. In particular, since the nodes are typically connected through a common channel without the need of multi-hop communication, it turns out that a timely communication mainly depends on how the nodes access the channel. Even when a multi-hop communication is needed, a timely message delivery is not feasible without the support of a predictable channel access mechanism, which is implemented in the Medium Access Control (MAC) sub-layer of the communication stack.
There exist several MAC protocols designed for providing a timely communication among distributed nodes, mainly in the factory communication domain. One of the most effective solutions is given by token passing protocols, which have some nice characteristics that make them suitable for real applications. For instance, because of the token passing mechanism, the nodes do not need to be synchronized and they have an implicit bandwidth reclaiming mechanism that allows other nodes to exploit the unused bandwidth. Moreover, such protocols can serve both real-time and best-effort (non real-time) traffic.
Among token passing protocols, the timed token policy is a channel scheduling approach first proposed by Grow in (Grow, 1982). Since then, it has received a substantial attention and several relevant results have been derived (Zhang et al., 2004), which make timed token protocols suitable for the real-time communication in industrial applications. Timed token policies are used as channel access mechanism in several standard protocols such as, for instance, PROFIBUS (Profibus, 1996) and FDDI (FDDI, 1987). However, the application domain of the timed token policy is not restricted to the cited communication standards, and some examples on their use can be found in (Lenzini et al., 2004) and (Cicconelli et al., 2007).
To improve the ability of timed token protocols of managing real-time traffic, Shin and Zheng (Shin & Zheng, 1995) proposed a modification of the timed token protocol, which can guarantee a greater bandwidth for the real-time traffic with respect to the classic timed token protocol; however, under certain conditions, it cannot manage the best-effort traffic (Franchino et al., 2008). The Budget Sharing Token (*BuST*) protocol has been proposed as an improvement with respect to existing timed token protocols (Franchino et al., 2008). In

particular, *BuST* improves the bandwidth guaranteed for the real-time traffic with respect to the timed token protocol, and it can manage best-effort traffic in those situations where the modified timed token protocol fails on serving this kind of traffic.
This chapter introduces a few token passing protocols, presenting their characteristics and illustrating the main results available in the literature. The *BuST* protocol is discussed in detail, illustrating its main advantages by means of analytic and simulation comparisons. New and well known properties of the protocols for managing both hard real-time and best-effort traffic will be analyzed, describing drawbacks and strengths of each protocol. The analysis is carried out considering the main budget allocation schemes available in the literature, thus contributing to the comparative characterization of the several schemes nowadays available.

2. Network Model

The communication network is composed by n nodes sharing a common medium, e.g. a bus, where each node can transmit both real-time and best-effort traffic. The former kind of traffic is modelled by assigning each node i a synchronous message stream S_i, which is described by three parameters (C_i, D_i, T_i), where:

- C_i is maximum amount of time necessary to transmit a message generated in the stream S_i. This includes the time to transmit both the message payload and the message headers/footers;
- D_i is the relative deadline of a message generated by stream S_i, that is, the maximum amount of time that a message can wait in transmission queue before its transmission is completed. Hence, the transmission of the *j-th* message, which is queued at time $t_{i,j}$, must be completed no later than its absolute deadline $d_i = t_{i,j} + D_i$.
- T_i is the generation period of messages in stream S_i. If the *j-th* message in the stream S_i is put in the transmission queue at time $t_{i,j}$, then the *(j+1)-th* will arrive in the transmission queue at time $t_{i,j+1} = t_{i,j} + T_i$.

Without loss of generality, only a stream per node it is considered, since the case of a network with more streams per node can be represented with a logical equivalent one with a stream per logical node, as showed in (Agrawal et al., 1994). We consider that each node i is assigned a bandwidth H_i, also called time budget, used to bound the transmission of the node.
The channel utilization of each message in stream S_i is:

$$U_i^S = \frac{C_i}{\min(T_i, D_i)} \tag{1}$$

The total channel utilization of a periodic stream set is then:

$$U^S = \sum_{i=1}^{n} U_i^S \ . \tag{2}$$

which measures the channel bandwidth utilized by the real-time (periodic) traffic.
Before describing the protocols, the following definitions are needed:

Definition 1. τ *is the time needed to transmit the token between nodes, including the overhead introduced by the protocol.*

Definition 2. The *Target Token Rotation Time* (*TTRT*) *represents the expected time needed by the token to complete an entire round-trip of the network.*

Any assignment of the time budgets H_i must satisfy the following two constraints:

Definition 3. (*Protocol Constraint*) *The total bandwidth allocated to the nodes, during one complete token rotation, must be less than the available network bandwidth, that is,*

$$\frac{\sum_{i=1}^{n} H_i}{TTRT} \leq 1 - \frac{\tau}{TTRT} \tag{3}$$

The Protocol Constraint is necessary to ensure a stable operation of the protocols.
Definition 4. (*Deadline Constraint*) *If $s_{i,j}$ denotes the time at which the transmission of the j-th message in stream S_i is completed, the deadline constraint requires that for i=1, …, n, and j=1, 2, …,*

$$s_{i,j} \leq t_{i,j} + D_i \tag{4}$$

where $t_{i,j}$ is the message arrival time and D_i is its relative deadline.
Meeting the Deadline Constraint ensures that every periodic message is transmitted before its absolute deadline. Note that in Inequality (4), while $t_{i,j}$ and D_i are defined by the application, $s_{i,j}$ depends on the bandwidth (budget) allocation and on the *TTRT* value.
Definition 5. *A stream set $\Gamma=\{S_1, S_2, \ldots, S_n\}$ is said to be feasible or schedulable when both the Deadline Constraint and the Protocol Constraint are met.*
To test the Protocol Constraint it is sufficient to check whether the sum of the budgets are not greater than the *TTRT* minus the overhead τ. However, testing the Deadline Constraint may be much more complicated. The following method is often used for testing whether the Deadline Constraint is met:
Let X_i be the minimum amount of time available for node i to transmit its *j-th* message during the time interval $(t_{i,j}, t_{i,j} + D_i]$, then for a message stream set with message deadlines not greater than periods, the Deadline Constraint can be satisfied if and only if for all i, $i=1,2,\ldots,n$, $X_i \geq C_i$.

3. Timed Token Protocols

The timed token protocol (*TTP*) is a basic channel access technique, namely a MAC protocol, which can be used to manage real-time traffic while guaranteeing a fair sharing of the unused bandwidth to the best-effort traffic. In timed token approaches, a token travels between nodes in a circular fashion and each node can transmit only when it possesses the token. An important parameter is the Target Token Rotation Time (*TTRT*), which represents the expected time needed by the token to complete an entire round-trip of the network. Each node i has an associated time budget H_i; whenever a node receives the token, it can transmit

its real-time messages for a time no greater than H_i. It can then transmit its best-effort messages if the time elapsed since the previous token arrival to the same node is less than the value of *TTRT*, that is, only if the token arrives earlier than expected. Figure 1, shows a typical timed token based network where 4 nodes sharing a common channel are arranged in a logical ring, as far as the token passing mechanism is concerned.

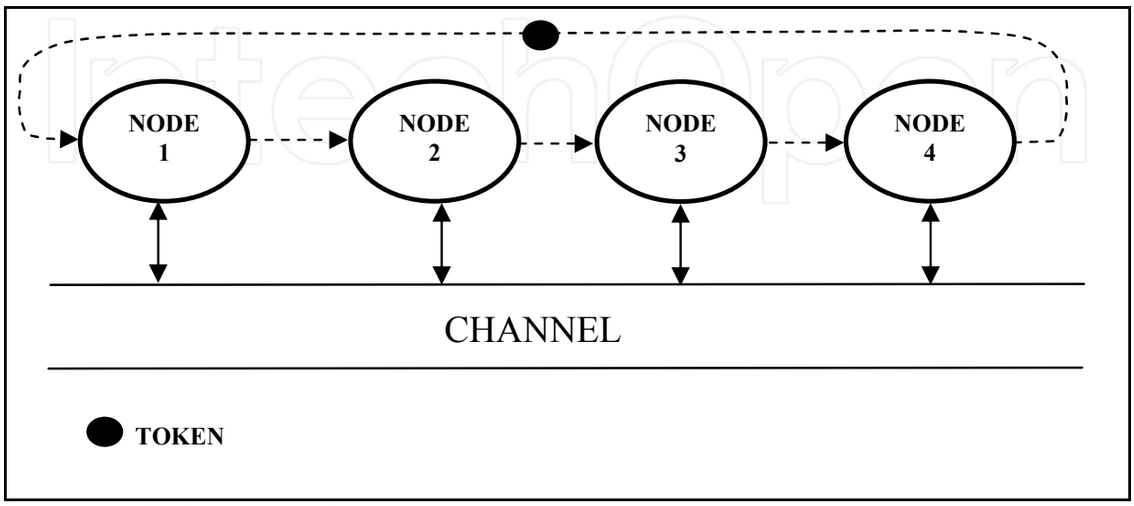

Fig. 1. Timed token network

3.1 *TTP* operation rules

To better understand the *TTP* operation, the protocol channel access rules are detailed below:

- During the network start up, each node *i* declares a *TTRT* value equal to one half of the deadline D_i related to its message stream S_i. The minimum declared value is chosen as *TTRT*, and each node *i* is then assigned a time budget H_i that depends on *TTRT*.
- Each node uses 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 best-effort traffic. When *TRT* reaches *TTRT*, it is reset to 0 and the token is signed as "late" by incrementing the node's late counter L_c by one. To initialize the timers and L_c, no messages are sent during the first token rotation after the ring initialization.
- Only the node holding the token can transmit messages. Transmission is controlled by the timers, but an in-progress transmission of a single packet is not interrupted until its completion. When node *i* gets the token, it performs the following operations:
    - If $L_c > 0$, it sets $L_c = L_c - 1$ and *THT* = *TTRT*. Otherwise, *THT* = *TRT* and *TRT* = 0.
    - If node *i* has synchronous packets, it transmits them for a time no greater than H_i.
- If node *i* has best-effort packets, it transmits them until *THT* counts up to *TTRT*, or until all the asynchronous traffic is sent, which ever comes first.

- Node i passes the token to station (i +1) mod (n).

The main drawback of *TTP* is the inability of guaranteeing the total available bandwidth for the real-time traffic. As Johnson and Sevciks (Johnson & Sevciks, 1987) showed, the average interval between two consecutive visits of the token at the same node, namely the average token rotation time, does not exceed *TTRT* and the maximum rotation time does not exceed 2*TTRT*. Thus, if D is the minimum deadline among stream deadlines, i.e. $D = min(D_i)$, it turns out that $TTRT = D/2$. From the Protocol Constraint it follows that:

$$\sum_{i=1}^{n} H_i \leq TTRT - \tau = \frac{D}{2} - \tau \tag{5}$$

In the time interval defined by D, the bandwidth available for the real-time traffic can be obtained dividing the above equation by D:

$$\frac{\sum_{i=1}^{n} H_i}{D} \leq \frac{1}{2} - \frac{\tau}{D} \tag{6}$$

Since the total available bandwidth is 1 – τ/TTRT, the *TTP* can guarantee at most half of the total available bandwidth for the real-time traffic.

3.2 Modified *TTP* operation rules

To improve the bandwidth guaranteed for the real-time traffic, Shin (Shin & Zheng, 1995) proposed to modify the timed token rules, limiting the maximum time available for best-effort traffic to T_B^{MAX}, where:

$$T_B^{MAX} = TTRT - \sum_{i=1}^{n} H_i - \tau \tag{7}$$

Notice that, under *TTP*, the maximum bandwidth allocated for best-effort messages is $T_B^{MAX} = TTRT - \tau$.

In *(Chan et al., 1997)*, the authors showed that the maximum token rotation time for the Modified Timed Token Protocol (*MTTP*) is bounded by *TTRT*. This means that, under MTTP it is possible to select a value for *TTRT* no greater than the minimum deadline D. Hence, being $TTRT \leq D$, from the Protocol Constraint it follows that:

$$\sum_{i=1}^{n} H_i \leq D - \tau \tag{8}$$

and dividing by D, it gives:

$$\frac{\sum_{i=1}^{n} H_i}{D} \leq 1 - \frac{\tau}{D} \tag{9}$$

That is, *MTTP* can guarantee all the available bandwidth for the real-time traffic.
By defining $T_S = \sum H_i$, to guarantee that $T_B^{MAX} = TTRT - T_S - \tau$, *MTTP* can be defined as follows:

1. A new token rotation time is defined as $TTRT_n = TTRT - T_S$, and used instead of *TTRT*;
2. The counting of a node's token rotation timer (*TRT*) is stopped when a real-time message is being delivered by the node.

The full details of the protocol rules are given below:

- During the network start up, each node i declares a *TTRT* value equal to the deadline D_i related to its message stream S_i. The minimum declared value is chosen as *TTRT*. Each node i is assigned a time budget H_i that depends on *TTRT*, and it sets $TTRT_n = TTRT - T_S$.
- Each node uses two timers, the token holding timer (*THT*) and the token-rotation-timer (*TRT*). The *TRT* counter increases only when the node is not transmitting real-time traffic, whereas the *THT* only increases when the node is delivering best-effort traffic.
- Only the node holding the token can transmit messages. Transmission is controlled by the timers, but an in-progress transmission of a single packet is not interrupted until its completion. When node i gets the token, it performs the following operations:
  - If $L_c > 0$, it sets $L_c = L_c - 1$ and $THT = TTRT$. Otherwise, $THT = TRT$ and $TRT = 0$.
  - If node i has synchronous packets, it transmits them for a time no greater than H_i.
- If node i has best-effort packets, it transmits them until *THT* counts up to *TTRT*, or until all the asynchronous traffic is sent, which ever comes first.
- Node i passes the token to station $(i+1) \bmod (n)$.

4. The *BuST* protocol

The *BuST* protocol has been devised to improve the ability of *TTP* in managing real-time traffic, and to overcome the problems *MTTP* can have in managing best-effort traffic (see Section 7).
Like timed token protocols, the *BuST* protocol assigns each node a time budget H_i for transmitting its real-time traffic. When a node receives the token, it can transmit the associated real-time traffic for a time no greater than the corresponding budget. The main difference with respect to *TTP* and *MTTP* concerns the best-effort traffic service. Under *TTP*, when the token arrives early, the node can transmit best-effort traffic for a time no greater than $T_A = TTRT - \tau - T_{LRT}$, where T_{LRT} is the time spent in the last round-trip of the token. Using *MTTP,* a node does the same, but with $T_A = TTRT - \tau - \sum H_i$. With *BuST*, a node can deliver non real-time traffic each time it gets the token, early or not, using the spare budget left by real-time messages. If H_i^{cons} is the budget consumed by node i to deliver periodic traffic, then it can send best-effort traffic for a time no greater than $T_{A_i} = H_i - H_i^{cons}$, even if

the token is not early. Observe that, *TTP* and *MTTP* can deliver best-effort traffic only when the token is early, that is, when $T_{LRT} < TTRT - \tau$.
The following rules specify the the *BuST* protocol in detail:

- During the network initialization phase, each node i declares a *TTRT* value equal to the deadline D_i of its periodic message stream. The minimum declared value is selected as *TTRT*. Each node i is assigned a time budget H_i that depends on *TTRT*.
- Each node has one timer, the token holding-rotation timer *THRT*. The *THRT* counter always increases. To initialize the timers, no messages are sent during the first token rotation.
- Only the node having the token can transmit messages. The transmission is controlled by *THRT*, but an in-progress transmission of a single packet is not interrupted until its completion. When node i gets the token, it performs the following operations:
    - It sets *THRT* = 0.
    - If node i has real-time packets, it transmits them until *THRT* counts up to H_i, or until all the real-time traffic is sent, whichever comes first.
    - If node i has best-effort packets, it transmits them until *THRT* counts up to H_i, or until all the best-effort traffic is sent, which ever comes first.
    - If a real-time message becomes ready during the transmission of best-effort packets, and $THRT < H_i$, the transmission is stopped and the node starts delivering the real-time traffic until *THRT* counts up to H_i, or until all the periodic traffic is sent, whichever comes first.
    - If node i completes the transmission of the periodic traffic without entirely consuming its budget, i.e. $THRT < H_i$, it starts transmitting its non real-time traffic, if any, until *THRT* counts up to H_i, or until all the best-effort traffic is sent, which ever comes first. Note that, in this case, the transmission is not stopped even if a real-time message becomes ready.
    - Node i passes the token to station $(i+1) \bmod (n)$.

The overhead generated when a best-effort transmission is interrupted can be easily accounted in τ. The same observation can be done for the overhead generated when an in-progress packet transmission is not interrupted until its completion. This also holds for *TTP* and *MTTP*.
As we can note from the protocol rules, under *BuST*, any node i exploits its time budget H_i to deliver both real-time and non real-time messages. When compared to *TTP*, *BuST* improves (as *MTTP*) the bandwidth available for real-time messages and halves the bandwidth lost due to the protocol overhead. In addition, *BuST* is able to deliver best-effort traffic also in those cases in which *MTTP* fails, as it will be shown in Section 7.
As a final remark, the implementation of *BuST* only requires one timer, instead of the two timers needed by *TTP* and *MTTP*. This can be useful when *BuST* is adopted in small embedded systems, where resources (e.g., hardware timers) are scarce.

5. Time properties

In this section, we introduce the main time properties of the protocols under analysis. These properties are the basis to understand the protocols timing behaviour, to verify if a given periodic stream set $\boldsymbol{M}=\{S_1, S_2, \ldots, S_n\}$ is feasible, i.e. both the Protocol and the Deadline

Constraints are met. Moreover, the results showed in the following can be used to analyse the protocols performance under the budget allocation schemes introduced in the next sections.
Lemmas 5.1, 5.2 and 5.3 provide an upper bound on the maximum transmission time for a real-time message under the *BuST*, *MTTP* and *MTTP* protocols.

Lemma 5.1 *Under the BuST protocol, for all budget allocation schemes, if $T_i \geq TTRT$, i=1 ,…,n, it holds:*

$$\forall i,j : s_{i,j} \leq t_{i,j} + \left\lceil \frac{C_i}{H_i} \right\rceil \left(\sum_{k=1}^{n} H_k + \tau \right) \tag{10}$$

Proof. See (Franchino et al., 2007). □

Lemma 5.2 *Under the TTP protocol, for all budget allocation schemes, if $T_i \geq 2TTRT$, i=1 ,…,n, it holds:*

$$\forall i,j : s_{i,j} \leq t_{i,j} + \left(\left\lceil \frac{C_i}{H_i} \right\rceil + 1 \right) TTRT + C_i - \left\lceil \frac{C_i}{H_i} \right\rceil H_i \tag{11}$$

Proof. See (Franchino et al., 2007).

□

Lemma 5.3 *Under the MTTP protocol, for all budget allocation schemes, if $T_i \geq TTRT$, i=1 ,…,n, it holds:*

$$\forall i,j : s_{i,j} \leq t_{i,j} + \left\lceil \frac{C_i}{H_i} \right\rceil TTRT + C_i - \left\lceil \frac{C_i}{H_i} \right\rceil H_i \tag{12}$$

Proof. See (Franchino et al., 2007).

□

The results and proofs in (Chen & Zhao, 1992) and (Chan et al., 1997) have been used as basis to derive the properties of *BuST* shown in the following. For comparison, the same type of results for *TPP* and *MTTP*, available in literature, are reported as well.
Theorem 5.1 gives the upper bound between any two consecutive visits of the token at the same node.

Theorem 5.1 *Under the BuST protocol, for any i=1 ,…,n, and for any integer $l > 0$ it turns out that:*

$$t_i(l+1) - t_i(l) \leq \sum_{j=1}^{n} H_j + \tau \tag{13}$$

where $t_i(l)$ is the time the token makes the l^{-th} visit at node i.

Proof. Since a node *i* can use only its time budget H_i to transmit both real-time and best-effort traffic, the length of interval $[t_i(l), t_i(l+1)]$ is equal to the sum of:

1. time for real-time traffic transmission: $t_{rt} \le \sum_{j=1}^{n} H_j$;
2. time for non real-time traffic transmission: $t_{nrt} \le \sum_{j=1}^{n} H_j - t_{rt}$;
3. time due to protocol overhead and token passing: τ.

Thus,

$$t_i(l+1) - t_i(l) = t_{rt} + t_{nrt} + \tau \le t_{rt} + \sum_{j=1}^{n} H_j - t_{rt} + \tau = \sum_{j=1}^{n} H_j + \tau \cdot$$

□

The following corollary, generalizes the last theorem providing the upper bound on the time elapsed between *v* consecutive token arrivals at the same node.

Corollary 5.1 *Under the BuST protocol, for any, i=1 ,…,n, and for any integer l > 0 and v > 0 it turns out that:*

$$t_i(l+v) - t_i(l) \le v \cdot \left(\sum_{j=1}^{n} H_j + \tau \right) \tag{14}$$

where $t_i(l)$ is the time the token makes the l-th visit at node i.

Proof. By Theorem 5.1, it follows that:

$$t_i(l+v) - t_i(l) = [t_i(l+1) - t_i(l)] + [t_i(l+2) - t_i(l+1)] + \ldots + [t_i(l+v) - t_i(l+v-1)]$$
$$\le \sum_{j=1}^{n} H_j + \tau + \sum_{j=1}^{n} H_j + \tau + \ldots + \sum_{j=1}^{n} H_j + \tau = v \cdot \left(\sum_{j=1}^{n} H_j + \tau \right) \cdot$$

□

The following result, gives the upper bound between two consecutive token visits at the same node with the timed token protocol.

Theorem 5.2 *Under the TTP protocol, for any, i=1 ,…,n, and for any integer l > 0 it turns out that:*

$$t_i(l+1) - t_i(l) \le TTRT + \sum_{j=1}^{n} H_j + \tau \le 2 \cdot TTRT \tag{15}$$

Proof. See (Sevcik & Johnson, 1987). □

The next corollary, provides the upper bound on the time elapsed between v consecutive token visits at same node.

Corollary 5.2 *Under the TTP protocol, for any i=1 ,…,n, and for any integer l > 0 and v > 0 it turns out that:*

$$t_i(l+v) - t_i(l) \le v \cdot TTRT + \sum_{j=1, j\neq i}^{n} H_j + \tau \tag{16}$$

where $t_i(l)$ is the time the token makes the l-th visit at node i.

Proof. See (Chen & Zhao, 1992). □

The bound between two consecutive token arrivals at the same node, under the modified timed token protocol, is provided by the following theorem.

Theorem 5.3 *Under the MTTP protocol, for any i=1 ,…,n, and for any integer l > 0 it turns out that:*

$$t_i(l+1) - t_i(l) \le TTRT \tag{17}$$

Proof. See (Chan et al., 1997). □

The corollary below, generalizes the last theorem giving the upper bound between v consecutive token vistis at the same node.

Corollary 5.3 *Under the MTTP protocol, for any i=1 ,…,n, and for any integer l > 0 and v > 0 it turns out that:*

$$t_i(l+v) - t_i(l) \le v \cdot TTRT \tag{18}$$

where $t_i(l)$ is the time the token makes the l^{-th} visit at node i.

Proof. See (Chan et al., 1997). □

The following lemmas, give the minimum amount of time available for node i to transmit its real-time traffic before the deadline of each message. Notice that, for simplicity's sake we consider a deadline $D_i = T_i$. When the deadline D_i is less than the period T_i, it is sufficient to replace T_i with D_i in the following results.

Lemma 5.3 *Under the BuST protocol, for any i=1,...,n, if at time t a periodic message with period T_i arrives at node i, then in time interval $(t,t+T_i]$ the minimum amount of time X_i available for node i to transmit the message is:*

$$X_i \le \left\lfloor \frac{T_i}{\sum_{j=1}^{n} H_j + \tau} \right\rfloor H_i + \max\left(0, H_i - \delta_i\right) \tag{19}$$

$$where\ \delta_i = \left\lfloor \frac{T_i}{\sum_{j=1}^{n} H_j + \tau} \right\rfloor \left(\sum_{j=1}^{n} H_j + \tau \right) - T_i .$$

Proof. Suppose t the time at which a new message is ready at node i. In the worst case, the token has just left node i when the new message is ready. As a consequence of Theorem 5.1, in time interval $I_i = [t,\ t + \sum_{j=1}^{n} H_j + \tau]$ node i receives the token at least once. Let $t + t_1$ be the time at which node i receives the token since t, it turns out that $0 < t_1 \le \sum_{j=1}^{n} H_j - H_i + \tau$. Hence, $H_i \le \sum_{j=1}^{n} H_j + \tau - t_1$, that is, the time available for node i to transmit its real-time traffic in I_i is H_i. Thus, by Corollary 5.1, it turns out that in a time interval $I_i = [t,\ m \cdot (t + \sum_{j=1}^{n} H_j + \tau)]$ node i has a minimum amount of time, to deliver its real-time traffic, equal to $m \cdot H_i$.

If $\Delta_i = T_i / (\sum_{j=1}^{n} H_j + \tau)$ and $m = \lfloor \Delta_i \rfloor$, two cases are possible:

1. Δ_i is an integer, such that $\delta_i = 0$ and $I_m = \left[t, t + m \cdot \left(\sum_{j=1}^{n} H_j + \tau \right) \right] = [t, t + T_i]$, then it follows that $X_i = \left\lfloor \frac{T_i}{\sum_{j=1}^{n} H_j + \tau} \right\rfloor H_i$.

2. Δ_i is not an integer, such that $\delta_i > 0$. In the worst case, the (m+1)-th token's arrival at node i may be as late as

$$t + m \cdot \left(\sum_{j=1}^{n} H_j + \tau \right) + \sum_{j=1}^{n} H_j + \tau + H_i ,$$

then, this token arrival will be not more than $t + T_i$ if:

$$\sum_{j=1}^{n} H_j + \tau - H_i \le T_i + m \cdot \left(\sum_{j=1}^{n} H_j + \tau \right) = \sum_{j=1}^{n} H_j + \tau - \delta_i .$$

In this case, the residual transmission time available for the real-time traffic is the left time until $t+T_i$, which is equal to $H_i - \delta_i$, that is:

$$X_i \le \left\lfloor \frac{T_i}{\sum_{j=1}^{n} H_j + \tau} \right\rfloor H_i + \max(0, H_i - \delta_i)$$

□

Lemma 5.4 *Under the TTP protocol, for any i=1,...,n, if at time t a periodic message with period T_i arrives at node i, then in time interval $(t,t+T_i]$ the minimum amount of time X_i available for node i to transmit the message is:*

$$X_i \le \left\lfloor \frac{T_i}{TTRT} - 1 \right\rfloor H_i + \max\left(0, \min\left(\delta_i - \tau - \sum_{j=1, j\neq i} H_j, H_i\right)\right) \quad (20)$$

where $\delta_i = T_i - \left\lfloor \frac{T_i}{TTRT} \right\rfloor TTRT$.

Proof. See (Chen & Zhao, 1992). □

Lemma 5.5 *Under the MTTP protocol, for any i=1,...,n, if at time t a periodic message with period T_i arrives at node i, then in time interval $(t,t+T_i]$ the minimum amount of time X_i available for node i to transmit the message is:*

$$X_i \le \left\lfloor \frac{T_i}{TTRT} \right\rfloor H_i + \max(0, H_i - \delta_i) \quad (21)$$

where $\delta_i = \left\lfloor \frac{T_i}{TTRT} \right\rfloor TTRT - T_i$.

Proof. See (Chan et al., 1997). □

6. Performance analysis

The real-time guarantee of a stream set highly depends on the budget allocation scheme (*BAS*) adopted for budgets assignment given the stream set parameters.

In literature exist several budget allocation schemes provided for timed-token protocols. They are traditionally classified into global or local schemes, depending on whether they need global or local information to assign the budgets. Local information is, for instance, the node stream parameters. Global information is, for instance, the number of nodes and the total channel utilization U^S required by the streams.

In this work, the budget allocation schemes are classified as proposed in (Daoxu et al. 1998). They can be divided into two categories, depending on the way they assign the budgets. The first category is the set of the *TTRT-partitioning* schemes, where a scheme belonging to this

set assigns node budgets partitioning the time expected for the token to make a rotation of the network, i.e. $TTRT - \tau$. The *TTRT-partitioning* schemes analyzed in the following are shown in Table 1.
The second category of budget allocation schemes, is the set of C_i*-partitioning* schemes, where a scheme belonging to this class assigns each budget H_i partitioning the maximum time length, C_i , to send a message from the stream S_i among a certain number of token rotation cycles. The C_i*-partitioning* schemes analyzed in the following are shown in Table 2.

TTRT-partitioning schemes	Assignment rule
Proportional Allocation (*PA*)	$H_i = U_i(TTRT - \tau)$
Normalized Proportional Allocation (*NPA*)	$H_i = \frac{U_i}{U}(TTRT - \tau)$
Equal Partition Allocation (*EPA*)	$H_i = \frac{TTRT - \tau}{n}$

Table 1. *TTRT-partitioning* budget allocation schemes.

C_i*-partitioning* schemes	Assignment rule
Local Allocation (*LA*)	$H_i = \frac{C_i}{\left\lfloor \frac{T_i}{TTRT} - 1 \right\rfloor}$
Modified Local Allocation (*MLA*)	$H_i = \frac{C_i}{\left\lfloor \frac{T_i}{TTRT} \right\rfloor}$

Table 2. C_i*-partitioning* budget allocation schemes.

The budget allocation schemes showed in the tables above have been extensively analyzed in literature. For a complete survey, an interested reader can see (Zhang et al., 2004) for *TTP*; (Chan et al., 1997) and (Daoxu et al., 1998) for *MTTP*; (Franchino et al., 2007), (Franchino et al., 2008) and (Franchino et al., 2008a) for *BuST*.
In the following, we compare the schemes performance under the token passing protocols considered in this chapter.

6.1 Performance metric

For evaluating and comparing the performance of different budget allocation schemes several metrics have been proposed. One of the most widely adopted metric is the Worst Case Achievable Utilization (*WCAU*). The *WCAU* of a budget allocation scheme represents the largest utilization (U^*) of the network such that, for any real-time message set whose total network utilization is $U^S \leq U^*$, the budget allocation scheme can guarantee the timeliness of each single real-time message.
The *WCAU* test is useful to guarantee the feasibility of a periodic stream set when only an estimation of the amount of real-time traffic is known (i.e., the maximum time required to send a message) without requiring a detailed characterization of each single real-time message.

6.2 Budget allocation schemes analysis

In the following, we assume $D_i = T_i$ for all the streams, however, the same results can be derived for the case where $D_i < T_i$ by simply substituting D_i to T_i. To make the treatment clearer, when not differently specified, let $\beta_i = T_i/TTRT$, $\beta_{min} = min(T_i)/TTRT$ and $\alpha = \tau/TTRT$. Parameter α represents the bandwidth loss due to the overhead.

Table 3 summarizes the *WCAU*s of the budget allocation schemes considered in this work.

With the *PA* scheme, the *BuST* protocol is the only one having a *WCAU* greater than *0*, whereas with *TTP* and *MTTP* no stream set can be guaranteed.

With the *NPA* scheme, *BuST* and *MTTP* present the same WCAU which depends on β_{min}, hence the smaller *TTRT* the greater is the bandwidth that the protocols can guarantee for the real-time traffic. Since $\beta_{min} \leq 1$ the minimum *WCAU* for the *NPA*, under both *BuST* and *MTTP*, is $(1-\alpha)/2$. Instead, the *TTP* protocols with the *NPA* scheme presents a *WCAU* which does not depends on β_{min} and it is lower than that presented by the other protocols. It is worth observing that, under the *NPA* scheme, the *MTTP* protocol cannot serve best-effort traffic (Franchino et al., 2008) . Conversely, the *BuST* protocol presents the same *WCAU* of *MTTP* and can serve also best-effort traffic.

Under the *EPA* scheme, *BuST* and *MTTP* have a greater *WCAU* with respect to *TTP*. However, the *WCAU* of all the tree protocols is very poor and it depends on the number of nodes. As for the *NPA* scheme, also under the *EPA* scheme the *MTTP* protocol cannot serve non real-time traffic.

BuST, *MTTP* and *TTP* present the same *WCAU* under the *LA* scheme. As for the *NPA* scheme, the *WCAU* under the *LA* scheme depends on β_{min}, i.e. on the choice of *TTRT*. With the *LA* scheme $\beta_{min} \geq 2$, thus the minimum *WCAU* of this scheme is $(1-\alpha)/3$.

With the *MLA* scheme, *TTP* presents a null *WCAU*. Instead, both *BuST* and *MTTP* present a *WCAU* which depends on β_{min} and equivalent to that presented with the *NPA* scheme. As shown in (Franchino et al., 2008a) , with the *MLA* scheme *MTTP* can not serve best-effort traffic when $\sum_{i=1}^{n} H_i \geq TTRT + \tau$.

Allocation schemes	*TTP*	*MTTP*	*BuST*
PA	0	0	$\frac{1-3\alpha}{2(1-\alpha)}$
NPA	$\frac{1-\alpha}{3}$	$\left\lfloor \frac{\beta_{\min}}{\beta_{\min}+1} \right\rfloor (1-\alpha)$	$\left\lfloor \frac{\beta_{\min}}{\beta_{\min}+1} \right\rfloor (1-\alpha)$
ELA	$\frac{1-\alpha}{3n-(1-\alpha)}$	$\frac{1-\alpha}{2n-(1-\alpha)}$	$\frac{1-\alpha}{2n-(1-\alpha)}$
LA ($\beta_{min} \geq 2$)	$\left\lfloor \frac{\beta_{\min}-1}{\beta_{\min}+1} \right\rfloor (1-\alpha)$	$\left\lfloor \frac{\beta_{\min}}{\beta_{\min}+1} \right\rfloor (1-\alpha)$	$\left\lfloor \frac{\beta_{\min}}{\beta_{\min}+1} \right\rfloor (1-\alpha)$
MLA ($\beta_{min} \geq 1$)	0	$\left\lfloor \frac{\beta_{\min}}{\beta_{\min}+1} \right\rfloor (1-\alpha)$	$\left\lfloor \frac{\beta_{\min}}{\beta_{\min}+1} \right\rfloor (1-\alpha)$

Table 3. WCAU of the considered schemes.

7. Best-effort traffic service

So far, real-time streams service have been analyzed. This section briefly describes the best-effort service of the *BuST* protocol and its improvements with respect to *MTTP*.

As shown in Section 4, under *MTTP* the maximum time a node can use to deliver non real-time traffic is $T_A = TTRT - \sum_{i=1}^{n} H_i - \tau$. It can be observed that:

- For the PA scheme $T_A = TTRT - \sum_{i=1}^{n} U_i (TTRT - \tau) - \tau = (1 - U^S)(TTRT - \tau)$;
- For the *NPA* scheme $T_A = TTRT - \sum_{i=1}^{n} \frac{U_i^S}{U^S} (TTRT - \tau) - \tau = 0$;
- For the *EPA* scheme $T_A = TTRT - \sum_{i=1}^{n} \frac{TTRT - \tau}{n} = 0$;

This means that *MTTP* is not able to deliver best-effort traffic under both the *NPA* and the *EPA* schemes, while T_A depends on U^S using the *PA* scheme. Moreover, it can be observed that *MTTP* can not serve non real-time traffic when $\sum H_i \geq TTRT + \tau$, that is, when the Protocol Constraint is not met. To verify this last statement, it is sufficient to note that, as stated in Section 3.1, each time a node receives the token it can transmit non real-time traffic for a time no greater than T_A, and since $TTRT - \tau - \sum H_i < 0$, it follows that $T_A = 0$. This means that, when this last condition holds, *MTTP* can starve best effort traffic also with both the *LA* and the *MLA* schemes.
To analyze a worst-case scenario for non real-time service, we assume that each node receiving the token has always best-effort traffic to deliver. In this case, the total channel utilization of the network, including real-time and best-effort traffic, is equal to $1-\alpha$, i.e. the channel is fully utilized.
The following theorem provides the minimum bandwidth that a node i can exploit to deliver non real-time traffic under the *BuST* protocol.

Theorem 7.2 *Under the BuST protocol, a node i can guarantee for the non real-time traffic a minimum bandwidth U_i^{nrt}, which depends on the budget allocation scheme adopted. In particular:*

- with *PA*, $U_i^{nrt} = U_i^S \left(\frac{1}{U^S + \frac{\alpha}{1-\alpha}} - 1 \right)$;
- with *NPA*, $U_i^{nrt} = U_i^S \left(\frac{1-\alpha}{U^S} - 1 \right)$;
- with *EPA*, $U_i^{nrt} = \frac{1-\alpha}{n}$;

Proof. See (Franchino et al., 2007), (Franchino et al., 2008) and (Franchino et al., 2008a) .

□

We have shown that, under the *BuST* protocol, non real-time traffic at each node has a minimum bandwidth guaranteed. In addition, it is worth observing that, when not all nodes of the network have to send best-effort traffic, during the round trip of the token, the value of U_i^{nrt} can increase.

8. Simulation results

In this section the performance of *BuST*, *TTP* and *MTTP* is compared by simulation. The simulations have been performed through a discrete-event simulator written for this purpose in C language. The simulation scenario considers a network consisting of 10 nodes. Each node has a periodic stream with a relative deadline ranging from 10 *msec* to 100 *msec*. An infinite amount of non real-time traffic is assumed: this means that, every time a node receives the token, it has some non real-time traffic to deliver. As already stated in Section 7, in this case the total channel utilization $U^{TOT} = U^{nrt} + U^S$ is equal to the total available bandwidth 1-α. Node budgets are assigned using the *PA*, *NPA*, *LA*, and *MLA* budget allocation schemes.

Performance is evaluated through the Maximum Deadline Miss Ratio (*MDMR*), defined as the ratio between the number of messages that miss their deadline and the total number of generated messages. The *MDMR* is measured as a function of the real-time channel utilization U^S, ranging from 0.1 to 1.0 (step of 0.1). For each value of U^S, up to 1000 simulation runs are performed, and the *MDMR* is considered among the runs. A different stream set is generated each run. In particular, for each stream set the utilizations U^S have been generated randomly with a uniform distribution using the method proposed in (Bini & Buttazzo, 2004). For each value of U_i^S, a relative deadline D_i is generated randomly with a uniform distribution in the interval [10, 100] *msec*. Periods are assumed equal to deadlines, i.e for all i $T_i = D_i$. The message lengths C_i have been computed as $C_i = U_i^S D_i$. The overhead τ is assumed equal to 20 *μsec*.

In the simulations, the token is considered as never lost and no ring recovery process is implemented. In this way, a violation of the Protocol Constraint will not compromise the stability of the protocols.

For each scheme, three different scenarios have been considered in the simulations. One where $TTRT = min(D_i)$, one where $TTRT = min(D_i)/2$, and the last with $TTRT = min(D_i)$ and only real-time traffic in the network.

Note that, when there is only real-traffic *BuST*, *TTP* and *MTTP* operate in the same way, that is, they are in practice the same protocol. Therefore, as it will be shown in the following, in case of only real-time traffic all the three protocols present the same performance in terms of *MDMR*.

8.1 Results with the *PA* scheme

In this subsection the results with the *PA* scheme are analyzed. Figure 2 shows the *MDMR* when the *PA* scheme is used to assign the node budgets, and $TTRT = min(D_i)$.

As expected from the results showed in Section 6, as long as $U^S \leq 0.5$, with *BuST* all the messages meet their deadlines (*MDMR* = 0), while *TTP* and *MTTP* present a non-null

deadline miss ratio. For $U^S = 0.6$, *BuST* presents a very small *MDMR* (equal to 0.5%) that cannot be appreciated in the figure. For $U^S \geq 0.7$, *BuST* presents a significant *MDMR* which is about 76% when $U^S = 1$, that is, when the channel is over-utilized (remember that the available bandwidth is 1-α).
While *TTP* presents a significant *MDMR* for all values of U^S, *MTTP* presents a *MDMR* lower than that shown by *BuST* for $U^S \geq 0.7$.
Figure 3 shows the *MDMR* with $TTRT = min(D_i)/2$. Notice that, the performance of both *BuST* and *MTTP* improve as expected. Instead *TTP* still presents a very poor performance.
Figure 4 shows the *MDMR* when nodes have only real-time traffic to deliver and $TTRT = min(D_i)$. In this case, as said before, the three protocols operate in the same way, producing the same performance. As long as U^S is not greater than 0.5, there are not deadline misses. With $U^S = 0.6$ we have a very small *MDMR*, which is equal to 0.69%. For $U^S \geq 0.7$ the *MDMR* starts increasing significantly.

8.2 Results with the *NPA* scheme

Figure 5 reports the *MDMR* when the *NPA* scheme is used to assign node budgets and $TTRT = min(D_i)$.
As long as $U^S \leq 0.5$, *BuST* and *MTTP* have no deadline miss; for $U^S \geq 0.6$, they start experiencing deadline misses. It is worth noticing that, for $U^S = 0.6$, *BuST* presents a *MDMR* close to 0%, which is not appreciable in the figure.
TTP has deadline misses for all values of U^S; this is due to the fact that *TTP* requires that $TTRT \leq min(D_i)/2$ to work properly.
For $U^S \geq 0.9$, *MTTP* performs better than *BuST*. However, it is worth remembering that, under the *NPA* scheme, *MTTP* is not delivering non real-time traffic. This is the reason why it can provide a better service for real-time traffic.
Figure 6 shows the *MDMR* with $TTRT = min(D_i)/2$. Notice that, with a lower *TTRT,* the performance of all the three protocols improves, as expected by the theoretical analysis showed in the previous sections. In particular, *TTP* has no deadline miss as long as $U^S \leq 0.3$. For $U^S = 0.4$ and $U^S = 0.5$, *TTP* presents an *MDMR* close to 0%. For $U^S \geq 0.6$, *TTP* presents an *MDMR* significantly greater than 0%. Notice that, *BuST* and *MTTP* provide more or less the same performance, with the difference that *BuST* serves also non real-time traffic.
Figure 7 shows the *MDMR* when nodes have only real-time traffic to deliver and $TTRT = min(D_i)$.

8.3 Results with the *LA* scheme

Figure 8 shows the *MDMR* when the *LA* scheme is used to assign node budgets and $TTRT = min(D_i)/2$.
As long as $U^S \leq 0.8$, *MTTP* and *BuST* present a null *MDMR*. For $U^S = 0.9$, they present a *MDMR* not appreciable in the figure, which is less than 0.05% for both protocols. *TTP* presents a null *MDMR* as long as $U^S \leq 0.4$, and a *MDMR* less than 0.3% for $0.5 \leq U^S \leq 0.7$, which is not appreciable in the figure. For $U^S > 0.7$, *TTP* presents a *MDMR* significantly greater than *BuST* and *MTTP*.
Figure 9 shows the *MDMR* when the nodes have only real-time traffic to deliver. As it can be noted, the absence of non real-time traffic improves the protocols performance

considerably. In particular, as long as $U^S \leq 0.8$, the *MDMR* is null; for $U^S = 0.9$, the *MDMR* is still close to 0%, and when $U^S = 1$ the *MDMR* is about 7%.
Finally, it is important to notice that a message deadline miss is due to the Protocol Constraint violation. Furthermore, as highlighted in Section 7, when the Protocol Constraint is not met, or even if the sum of the budgets is equal to *TTRT* - τ, *MTTP* cannot serve non real-time traffic.

8.4 Results with the *MLA* scheme

Figure 10 shows the *MDMR* when the *MLA* scheme is used to assign the node budgets, and TTRT = $min(D_i)$.
As long as $U^S \leq 0.7$, *MTTP* and *BuST* present a null *MDMR*; for $U^S = 0.8$ and for $U^S = 0.9$, the MDMR is less than 1% for both protocols. Under *TTP*, the *MDMR* is non-null for all values of U^S. This is due to the fact that, as stated in Section 6, the Deadline Constraint cannot be satisfied when the *MLA* scheme is used under *TTP*.
Figure 11 shows the MDMR under the *MLA* scheme when *TTRT* = $min(D_i)/2$. For *BuST* and *MTTP*, as long as $U^S \leq 0.8$, the *MDMR* is null. When $U^S = 0.9$, the *MDMR* is not greater than 0.3%, hence is not appreciable in the figure, while when the channel is overloaded, i.e. $U^S = 1$, the *MDMR* is approximately equal to 15%.
Figure 12 depicts the *MDMR* when nodes have only real-time traffic to deliver and *TTRT*= $min(D_i)$. As said before, in this case, all the three protocols present the same performance, and the absence of non real-time traffic improves the protocols performance considerably. In particular, as long as $U^S \leq 0.8$, the *MDMR* is null or very small. For $U^S = 0.9$, the *MDMR* is less than 2%, and when $U^S = 1$ the *MDMR* is close to 5%.
As highlighted in Section 7, when the *MDMR* is not null it means that the Protocol Constraint is not met and, because of this, *MTTP* is not able to deliver non real-time traffic.

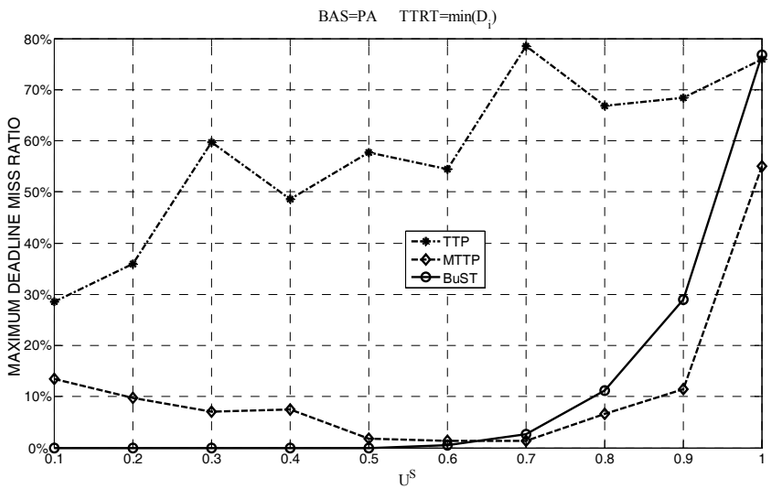

Fig. 2. Maximum Deadline Miss Ratio with the *PA* scheme.

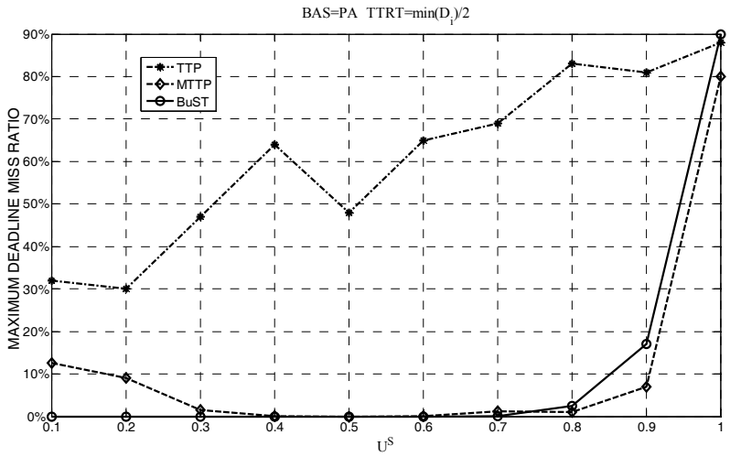

Fig. 3. Maximum Deadline Miss Ratio with the *PA* scheme when $TTRT = min(D_i)/2$.

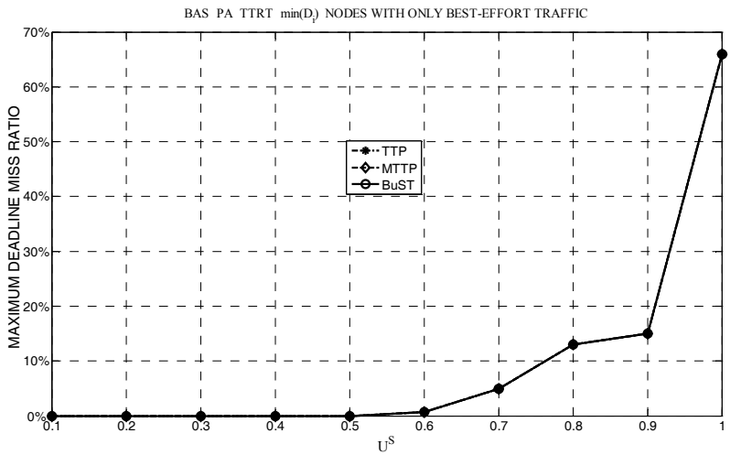

Fig. 4. Maximum Deadline Miss Ratio with the *PA* scheme with only best-effort traffic.

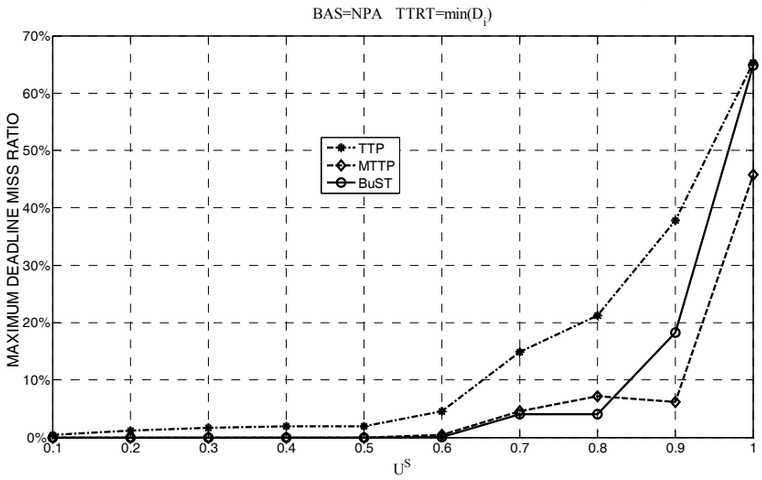

Fig. 5. Maximum Deadline Miss Ratio with the *NPA* scheme.

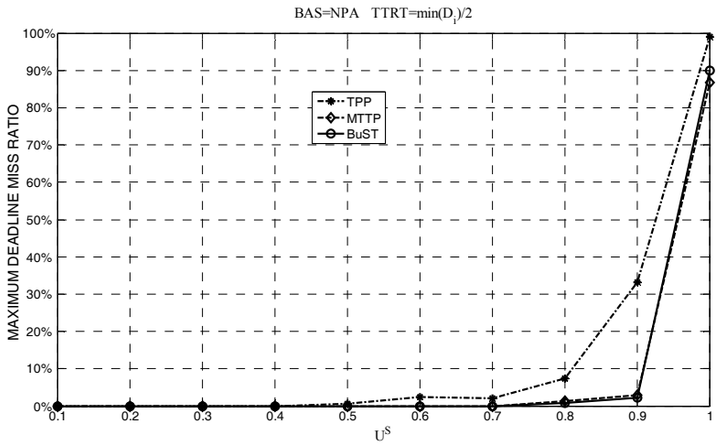

Fig. 6. Maximum Deadline Miss Ratio with the *NPA* scheme when $TTRT = min(D_i)/2$.

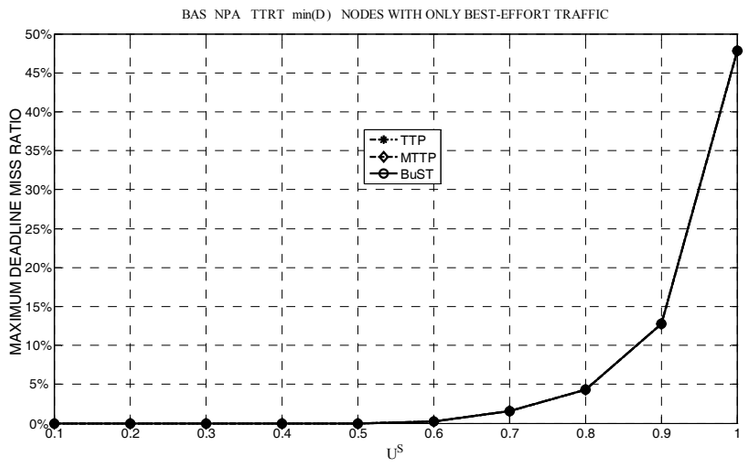

Fig. 7. Maximum Deadline Miss Ratio with the *NPA* scheme with only best-effort traffic.

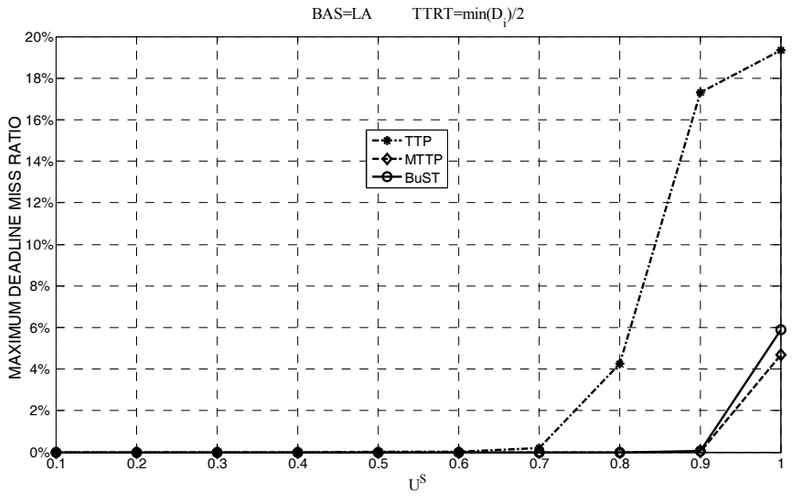

Fig. 8. Maximum Deadline Miss Ratio with the *LA* scheme.

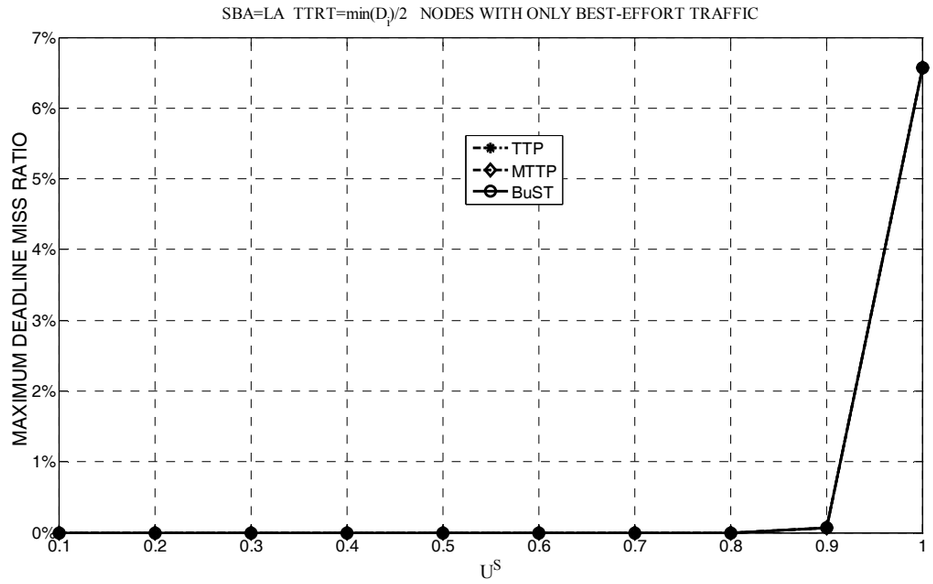

Fig. 9. Maximum Deadline Miss Ratio with the *LA* scheme with only best-effort traffic.

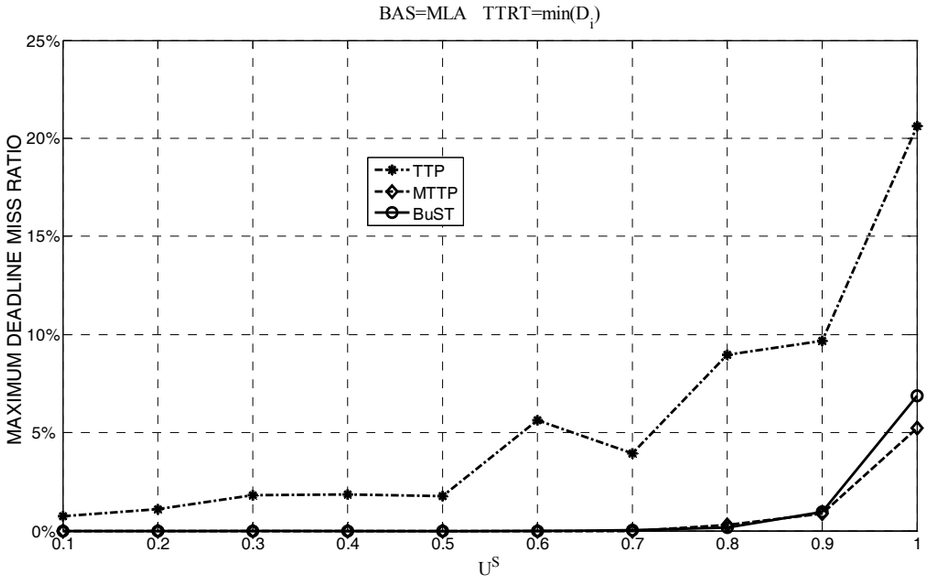

Fig. 10. Maximum Deadline Miss Ratio with the *MLA* scheme.

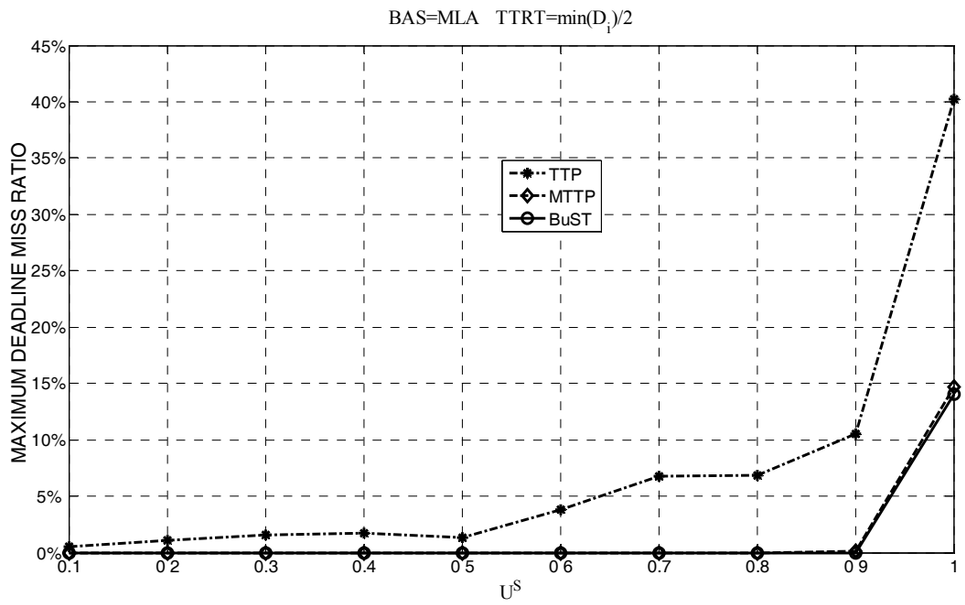

Fig. 11. Maximum Deadline Miss Ratio with the *MLA* scheme when $TTRT = min(D_i)/2$.

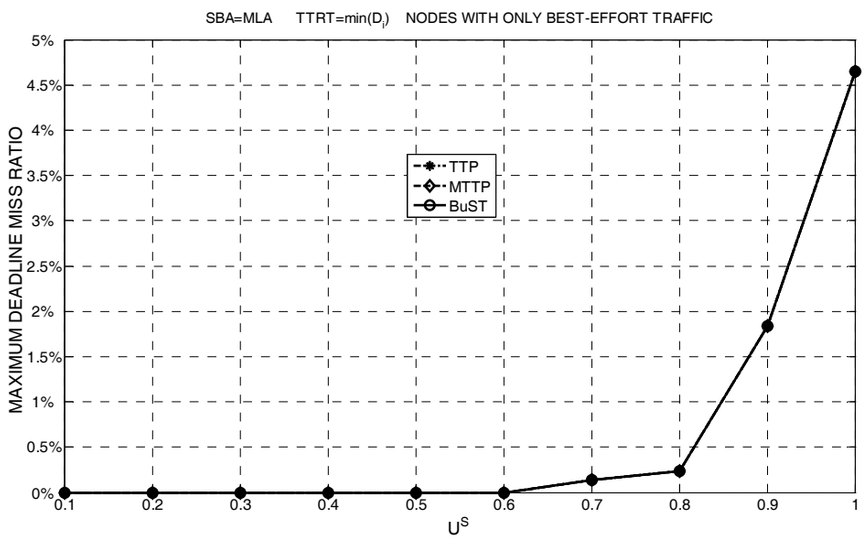

Fig. 12. Maximum Deadline Miss Ratio with the *MLA* scheme with only best-effort traffic.

9. Conclusions

In this work, we introduce and analyzed the *BuST* protocol in comparison with timed token and modified timed token protocols. All the three protocols are discussed in detail, showing their strengths and their drawbacks under different budget allocation schemes. New and well known time properties of the protocols are presented, together with their capability on managing both hard real-time and best-effort traffic. In particular, we have seen that both *BuST* and *MTTP* are superior on serving real-time traffic with respect to the traditional

timed token protocol. However, it has been shown that when both *NPA* and *EPA* schemes are used to assign the node budgets, *MTTP* can starve best-effort traffic. Conversely, the *BuST* protocol can serve also best-effort traffic under all the analyzed budget allocation schemes.
Future work is focused on extending the performance analysis of *BuST* with other allocation schemes available in literature.

10. References

Agrawal, G. ; Chen, B. ; Zhao, W. and Davari, S. (1994). Guaranteeing Synchronous Message Deadlines with the Timed Token Medium Access Protocol, *IEEE Transaction on* Computers, Vol. 43, No. 3, (March 1994), pp. 327-339, ISSN 0018-9340.

Bini, E. and Buttazzo, G. C. (2005). Measuring Performance of Schedulability Tests, *Real-Time Systems*, Vol. 30, No. 1-2 , (May 2005), pp. 129-154, ISSN 0922-6443.

Chan, E. ; Daoxu, C. ; Cao, J. ; and Lee, C. (1997). Timing Properties of the FDDI-M Medium Access Protocol. *The Computer Journal,* Vol. 40, No. 1, (1997), pp. 43-49, ISSN 0010-4620.

Chen, D. and Zhao, W. (1997). Properties of the Timed Token Protocol. *Technical Report 92-038,* Computer Science Dept., Texas A&M University, October 1992.

Daoxu, C. ; Chan, E. ; and Lee, V. C. S. (1998). Timing Properties of the FDDI-M Medium Access Protocol for a Class of Synchronous Budget Allocation Schemes. *Proceedings of 7-th Int. Conferance on Computer Communications and Networks*, *(ICCCN98)*, pp 825-832, Lafayette, Lousiana, USA, Oct. 1998.

Cicconetti, C. ; Lenzini, L. ; Mingozzi, E. & Stea, G. (2007). An Efficient Cross Layer Scheduler for Multimedia Traffic in Wireless Local Area Networks with IEEE 802.11e HCCA. *ACM Mobile Computing and Communication Review,* Vol. 11, No. 3, (July 2007), pp. 31-46, ISSN 1559-1662.

Fiber Distributed Data Interface (FDDI) (1987). Token Ring Medium Access Control (MAC), May 1987. ANSI Standard X3.139.

Franchino, G.; Buttazzo, G. C., Facchinetti, T. (2007). BuST: Budget Sharing Token Protocol for Hard Real-Time Communication, *Proceedings of 12th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA07)*, pp. 1278-1285, Patras (Greece), September 2007, IEEE.

Franchino, G.; Buttazzo, G. C., Facchinetti, T. (2008). Time Properties of the BuST Protocol under the NPA Budget Allocation Scheme, *Proceedings of the Conference on Design, Automation and Test in Europe (DATE 2008)*, pp. 1051-1056, Munich (Germany), March 2008, ACM.

Franchino, G.; Buttazzo, G. C., Facchinetti, T. (2008a). Properties of BuST and Timed Token Protocols in Managing Hard Real-Time Traffic, *Proceedings of 13th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA08)*, pp. 1205-1212, Hamburg (Germany), September 2008, IEEE.

Grow, R. M. (1982). A Timed Token Protocol for Local Area Networks. *Proceedings of Electro'82,* Paper 17/3, May 1982.

Lenzini, L. ; Mingozzi, E. & Stea, G. (2004). Design and Performance Analysis of the Generalized Timed Token Service Discipline. *IEEE Transaction on Computers,* Vol. 53, No. 7, (July 2004), pp. 879-891, ISSN 0018-9340.

Profibus (1996). General Purpose Field Communication System. European Standard EN50170, CENELEC, Vol. 2/3, Profibus, July 1996.

Sevcik, K. C. and Johnson, M. J. (1987). Cycle time properties of the FDDI token ring protocol. *IEEE Transaction Software Engineering,* Vol. SE-13, No. 3, (1987), pp. 376-385.

Shin, K. G. and Zheng, Q. (1995). FDDI-M: A Scheme to Double FDDI's Ability of Supporting Synchronous Traffic. *IEEE Transaction on Parallel and Distributed Systems,* Vol. 6, No. 11, (November 1995), pp. 1125-1131, ISSN 1045-9219.

Zhang, S. ; Burns, A. ; Chen, J. & Lee, E. (2004). Hard Real-Time communication with the Timed Token Protocol : Current State and challenging Problems. *Real-time Systems,* Vol. 27, No. 3 , (September 2004), pp. 271-295, ISSN 0922-6443.

Factory Automation

Edited by Javier Silvestre-Blanes

ISBN 978-953-307-024-7
Hard cover, 602 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

Factory automation has evolved significantly in the last few decades, and is today a complex, interdisciplinary, scientific area. In this book a selection of papers on topics related to factory automation is presented, covering a broad spectrum, so that the reader may become familiar with the various fields, and also study them in more depth where required. Within various chapters in this book, special attention is given to distributed applications and their use of networks, since it is one of the most relevant subjects in the evolution of factory automation. Different Medium Access Control and networks are analyzed, while Ethernet and Wireless networks are looked at in more detail, since they are among the hottest topics in recent research. Another important subject is everything concerning the increase in the complexity of factory automation, and the need for flexibility and interoperability. Finally the use of multi-agent systems, advanced control, formal methods, or the application in this field of RFID, are additional examples of the ideas and disciplines that experts around the world have analyzed in their work.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen