Wireless Channel Access Reservation for Embedded Real-time Systems

Dinesh Rajan[⋆], Christian Poellabauer[⋆], Xiaobo Sharon Hu[⋆], Liqiang Zhang[†], Kathleen Otten[⋆]
[⋆]Department of Computer Science and Engineering, University of Notre Dame
[†]Department of Computer and Information Sciences, Indiana University South Bend
{dpandiar, cpoellab, shu, kotten}@nd.edu, liqzhang@iusb.edu

ABSTRACT

Reservation-based channel access has been shown to be effective in providing Quality of Service (QoS) guarantees (e.g., timeliness) in wireless embedded real-time applications such as mobile media streaming and networked embedded control systems. While the QoS scheduling at the central authority (i.e., base station) has received extensive attention recently, the computation of resource requirements at each individual node has been widely ignored. An inappropriate resource requirement may lead to degraded support for real-time traffic and overprovisioning of scarce network resources. This work addresses this issue by presenting a strategy for nodes to determine minimal resource reservations that guarantee the real-time constraints of their network traffic. In addition, this paper examines the relationship between timeliness constraints of the traffic and resource requirements.

Categories and Subject Descriptors

C.3 [**Special-Purpose and Application-Based Systems**]: [Real-time and embedded systems]; D.4.4 [**Operating Systems**]: Communications Management—*Network communication*

General Terms

Design, Performance

Keywords

Wireless real-time systems, energy management, bandwidth reservation, packet scheduling, task scheduling

1. INTRODUCTION

Wireless embedded real-time systems are becoming prevalent with the continuous increase in streaming applications such as video/audio communications, mobile gaming, and wireless sensor and actuator networks. This has called for research efforts to enhance the support of timeliness and Quality of Service (QoS) in wirelessly networked embedded environments. Some recent efforts have led to the adoption of sophisticated protocols and mechanisms based on resource reservations to achieve the desired QoS objectives [4, 8, 10].

Techniques based on resource reservations allow resources to be negotiated and provisioned to nodes based on traffic requirements and resource availability. Channel access mechanisms based on resource reservations allow for contention-free accesses and thereby provide deterministic bounds on the delays experienced by the traffic streams. Therefore, such access mechanisms are ideally suited for providing *real-time services* in wireless environments. Access mechanisms based on reservations require each node to negotiate its required channel access duration and frequency of accesses based on its traffic constraints. However, the computation of such requirements has largely been ignored which has often resulted in poor real-time support, overprovisioning of valuable resources, and poor scalability.

The goal of this work is to develop a strategy for the computation of channel access reservation parameters such that a) the real-time constraints of each node's traffic are satisfied and b) resource reservations are minimized. The proposed formulations prevent a node from negotiating a greater share of the channel resources than is actually required. This prevents these resources from being overprovisioned, thereby providing better support for scalability.

In addition, the assignment of packet transmission deadlines that describe the timeliness requirements of the traffic is studied and their impact on resource reservations is investigated. Such an analysis is especially useful during system and application design where the range of feasible packet deadlines can be identified from the timeliness constraints and the actual deadline can then be chosen by considering its consequences in terms of resource requirements.

The contributions of this work are summarized as follows:

- formulation and identification of the *minimum worst-case* values for the channel access reservation parameters at each node that guarantee to meet the real-time requirements of its traffic;
- investigation of packet deadline assignment strategies, leading to the conclusion that increasing a packet deadline does not always lower resource requirements and the development of guidelines for their assignment if such flexibility exists.

It is important to note that we consider *managed* networks (as opposed to ad-hoc networks), where each node connected to a wireless base station (BS) executes the pro-

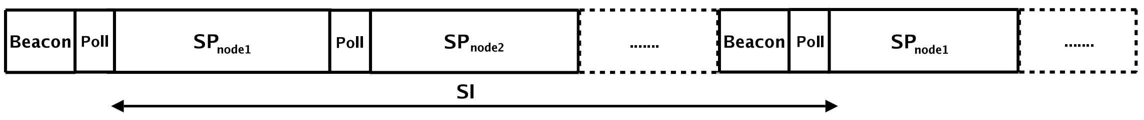


Figure 1: Description of channel access reservation parameters.

posed computation of required channel access parameters. Such an approach not only achieves more efficient utilization of channel resources but also reduces the overheads of the BS. Moreover, our work can also complement existing mechanisms at the BS [13]. To the best of our knowledge, this is the first such work to address the computation of reservation parameters at individual nodes (as opposed to the BS) in wireless real-time environments.

2. PRELIMINARIES

This section describes the channel access reservation mechanism, the periodic traffic model under consideration, the problems to be solved, and related work.

2.1 Reservation-based channel access

We briefly discuss reservation-based channel access since it forms the basis for the problem we intend to solve. Such a mechanism uses resource reservations to ensure contention-free accesses. This is achieved through a central authority at the BS that provisions and regulates the channel accesses by the individual nodes. Here, the BS takes control of the channel and starts polling each of the nodes in a pre-determined order (e.g., round-robin). On reception of a polling frame, a node gains access to the channel. The IEEE 802.11e standard [10] is an example which adopts the reservation-based channel access approach to enhance the QoS support for real-time applications in wireless environments.

Borrowing the terminology from the IEEE 802.11e standard [10], in a reservation-based channel access mechanism, each node is provided a *Service Period* (SP), during which the node has exclusive access to the wireless medium. Polling frames issued by the BS specify the start time and maximum duration of the SP allotted to a node. At the end of an SP for one node, the BS begins polling the next node in its schedule. The period of recurrence of the SPs is referred to as the *Service Interval (SI)*. The SP and SI parameters at each node must be negotiated with the BS based on the requirements of the node's expected real-time traffic. A scheduler at the BS is then responsible for deriving a schedule and provisioning the negotiated SP and SI to the respective nodes (shown in Figure 1). It is important to note that our work does not make any assumptions on the scheduling mechanism at the BS.

2.2 Traffic model

We consider a set of wireless nodes, $\{N_1, \cdots, N_r, \cdots, N_m\}$, each executing real-time applications and connected to each other via a BS. Each node executes a set of periodic tasks $\tau = \{\tau_1, \cdots, \tau_i, \cdots, \tau_n\}$ that generate real-time traffic. Each task τ_i has a period, p_i, and relative deadline, d_i, with $d_i \leq p_i$. These tasks are invoked periodically and the k^{th} invocation of task τ_i is referred to as job J_i^k. Examples of applications with such a periodic traffic model include streaming media application, sensor and actuator networks, embedded control systems, and other applications that periodically share sensor and control information.

Each J_i^k is assumed to generate a packet P_i^k that is part of a real-time stream generated by τ_i. A packet P_i^k is assumed to have a worst-case transmission time T_i. *Note that T_i can be derived from the worst-case packet size, the channel conditions, and the supported transmission rates.* For example, the latencies incurred during re-transmissions, which are required to successfully transmit data under the given error rates, can be included in T_i to account for error-prone channels. These latencies can be computed using the maximum number of re-transmission attempts [11]. In this work, we do not assume packets to be fragmented after their generation or packet transmissions to be preempted.

Each P_i^k is associated with a release time R_i^k and packet deadline D_i^k. R_i^k is the time when P_i^k is generated, placed into a packet queue, and ready for transmission. D_i^k denotes the time by which P_i^k must be transmitted from the corresponding node and it must satisfy the relationship $D_i^k \geq (R_i^k + T_i)$. Note that R_i^k and D_i^k are defined relative to the release time of the corresponding job. It is assumed that J_i^k can complete execution any time within its period and thus R_i^k can be anywhere in the duration between the start and end of the k^{th} period of task τ_i, i.e., in the interval (k-1*p_i, k*p_i). The packet deadline D_i^k is always assumed to be greater than or equal to the corresponding job deadline. Note that our work makes no assumptions on the task scheduling model. The tasks (and packets) can be released and executed based on any desired scheduling algorithm.

Finally, we make the simplifying assumption that SI_r is always chosen to be less than the periods of all traffic-generating tasks at node N_r. Such an assumption is reasonable as otherwise the probability that SP_r must be overprovisioned to meet packet deadlines becomes much larger.

2.3 Problem statement

From our earlier discussions, it is known that each node N_r is responsible for negotiating its required SP_r and SI_r values with the BS. The problem of concern is to compute SP_r and SI_r at node N_r such that the real-time requirements of the traffic generated by N_r are satisfied and the resource allocations are minimized. In this work, the term *bandwidth* (BW_r) is used to describe the requirements on the SP_r and SI_r parameters of node N_r. Formally this is given as

$$BW = \sum_{r=1}^{n} BW_r = \sum_{r=1}^{n} \frac{SP_r}{SI_r} \tag{1}$$

That is, the overall provisioned bandwidth (BW) is computed as the total of the bandwidth reservations (BW_r) required by each node N_r, where each reservation is expressed by a (SP_r, SI_r) pair. Thus in order to minimize BW, each node has to carefully determine and negotiate its (SP_r, SI_r) in accordance with its traffic requirements. This challenge is formally defined in Problem 1.

Problem 1. *Given a set of packet-generating tasks at node N_r, determine an optimal (SP_r, SI_r) that satisfies the real-time constraints of N_r's traffic while minimizing BW_r.*

Additionally, we study the formulation of guidelines for the assignment of packet deadlines. The task of identifying packet deadlines has often proved challenging due to the lack of any directives illustrating the benefits and consequences of choosing a deadline. Typically they are assigned based on the timeliness constraints of end-to-end communications. This work investigates the trade-offs in the selection of packet deadlines with respect to resource requirements and proposes guidelines for their assignment.

Problem 2. *Given a range of feasible deadlines for a packet P_i^k, identify a deadline D_i^k that minimizes BW_r.*

2.4 Related work

Scheduling and schedulability analysis have been extensively studied in previous work, particularly for processing resources [7]. In networking environments, reservation-based mechanisms are becoming highly prominent in supporting delay and QoS-sensitive traffic. In this section, we discuss existing protocol standards and research efforts related to resource and channel access reservations.

2.4.1 IEEE 802.11e standard and HCCA mechanism

A well-known and recent wireless standard that offers channel access reservations is the IEEE 802.11e protocol [10]. The network model in our work utilizes the terminology and concepts of this protocol standard. For example, the definition of SP and SI is based on the channel access reservation parameters specified in this standard.

The IEEE 802.11e standard proposes a Hybrid Coordination Function (HCF) that provides both contention and contention-free channel accesses through two modes: the Enhanced Distributed Channel Access (EDCA) and the HCF Controlled Channel Access (HCCA) [12, 10]. The HCCA mode specifies a central control authority for the Hybrid Coordinator (HC), which typically exists at the BS, to regulate channel accesses by the different nodes and achieve contention-free accesses.

The HCCA mode utilizes the concept of traffic streams (TS) to differentiate between flows with different QoS requirements. Each TS of a node is provided with an individual transmission opportunity (TXOP). The frequency and length of the TXOPs are negotiated based on the QoS requirements of the individual streams. Also, the TXOPs provided for the streams at a node can be grouped together to form a continuous interval which corresponds to SP in our work. Similarly, the period of recurrence of these continuous intervals, which are also available for negotiation in IEEE 802.11e environments, corresponds to SI. It is important to note here that our work is also applicable to other similar reservation-based access mechanisms. This is because our work formulates the computation of the channel access durations and the access frequency of each node; parameters that are required for any bandwidth reservation approach.

2.4.2 Channel access and resource reservations

There have been several recent research efforts in providing resource reservation schemes for wireless environments. The work in [4, 8] present reservation-based channel access protocols for mobile and ad-hoc networks respectively. These efforts assume either cooperation among the communicating nodes [8] or an underlying cellular-IP architecture [4]. The work in [1] addresses the challenging problem of designing a polling-based QoS scheduler to achieve fairness among real-time flows and to maximize the overall system throughput simultaneously.

Several mechanisms have been proposed for TXOP allocation in the HCCA mode of IEEE 802.11e. Adaptive and effective QoS scheduling at the HC were employed in [5, 9, 13]. Feedback on the packet queue length [3] and its estimation based on traffic characteristics [2] were proposed to enhance the allocation mechanisms. In [6, 14], cross-layer optimizations across the MAC and application layers of the OSI stack were exploited to provide better QoS support for multimedia streams.

These related efforts focus on QoS scheduling at the network side (i.e., the base station). In comparison, our work proposes the computation of reservation parameters at the end systems or nodes and is thus independent of the mechanisms at the BS. This is a novel contribution since standards such as the IEEE 802.11e leave the implementation of the QoS scheduler at the BS to its manufacturers and users. Also, since the BS is a bottleneck resource in a wireless system, the computation of resource reservations at the end nodes can significantly lower overheads at the BS. Finally, an end-node based approach makes sense since each node knows the characteristics and parameters of its generated traffic and can therefore compute its resource requirements more efficiently than the BS (and it only needs to communicate its desired SP and SI values instead of detailed information describing its entire taskset).

3. A SINGLE REAL-TIME STREAM

This section discusses the formulation of (SP_r, SI_r) that minimizes the bandwidth requirements while satisfying the real-time requirements of the traffic generated by N_r. We first analyze the required (SP_r, SI_r) considering a single traffic-generating task. The conclusions from this analysis form the basis for determining these parameters in scenarios with multiple traffic-generating tasks. The analysis here leads to simple formulae for the worst-case SP_r required to satisfy the packet deadlines with a given SI_r. Finally, the optimal SI_r that requires the minimal SP_r is derived.

Since SP_r must be chosen such that no deadline violations occur under any circumstances, we first determine the worst-case scenarios that require the maximum value for SP_r in order to satisfy the given packet deadline. Consider task τ_i. Given that $SI_r \leq p_i$ (refer to end of Section 2.2), there is at most one packet to be transmitted in SI_r. We let this packet be P_i with release time R_i and deadline D_i (note that these simplified notations are used instead of P_i^k, R_i^k and D_i^k for the remainder of our analysis). Thus the interval (R_i, D_i) represents the time duration in which the released packet P_i is available for transmission before its deadline. We define this interval (R_i, D_i) as the **active window** of a packet. Figure 2(a) shows the active window for packet P_i released at the end of job J_i execution. Without loss of generality, we assume that SP_r always occurs at the beginning of SI_r (the assumption is valid since SI can be defined to be measured between the start of consecutive SPs and the SP always occurs at the same relative position in an SI). Intuitively, if SI_r starts right after P_i is released, SP_r can be simply set to T_i. However, if SI_r starts before P_i is released, some portion of SP_r would be wasted, i.e., $SP > T_i$. We refer to the portion of SP_r that is greater than T_i as the over-provisioning amount. The following lemma helps to determine the bound on such over-provisioning amount.

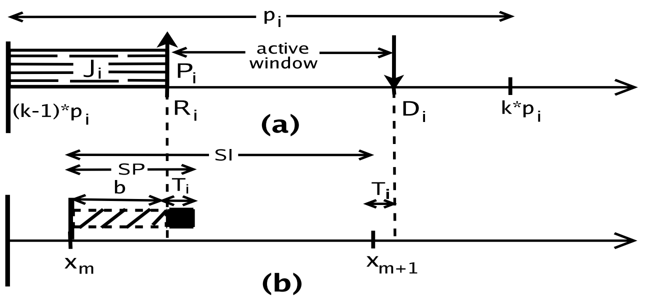


Figure 2: (a) Packet P_i generated by job J_i at R_i with deadline D_i and (b) Illustration of the over-provisioning amount b that is required in SP_r.

LEMMA 1. *Given a task τ_i and a service interval SI_r, the over-provisioning amount, denoted by b, (i.e., the amount required in addition to T_i to be provisioned for SP_r in order to transmit P_i by D_i) is bounded by B where*

$$B = \begin{cases} SI_r - (D_i - R_i) + T_i & \text{if } SI_r > (D_i - R_i) - T_i \\ 0 & \text{otherwise} \end{cases}$$

Proof: To prove the lemma, we observe the following facts: (i) at most one packet needs to be transmitted in SI_r in order to satisfy the deadline since $SI_r < p_i$; (ii) an over-provision for SP_r is only needed if a packet is released after the start of SP_r as otherwise the packet can be transmitted immediately upon release and no over-provision is required. We consider the two cases identified in the lemma separately.
Case 1. $SI_r > (D_i - R_i) - T_i$: We prove this case by contradiction, i.e., assuming $b > B$. Assume packet P_i is released in SI_r. Let x_m and x_{m+1} denote the start and end time of SI_r, respectively. That is, $x_m < R_i < x_{m+1}$. Figure 2(b) illustrates the over-provisioning amount b required when a packet is released such that $x_m < R_i < x_{m+1}$. (Note that if $R_i \le x_m$, no over-provision is needed, and if $R_i \ge x_{m+1}$, P_i will not be transmitted in SI_r.) Given that SP_r occurs at the beginning of SI_r and $B > 0$, we have

$$b = R_i - x_m > B = SI_r - (D_i - R_i) + T_i. \tag{2}$$

By regrouping the terms in (2) and noting that $SI_r = x_{m+1} - x_m$, we obtain $D_i - x_{m+1} > T_i$. It follows that packet P_i can be postponed for transmission at or after x_{m+1} without violating its deadline. Based on fact (i), P_i is the only packet released in SI_r, and thus no provision is needed in SI_r. That is, $b = 0$, which contradicts the hypothesis of $b > B$.
Case 2. $SI_r \le (D_i - R_i) - T_i$: By regrouping the given case condition and substituting $x_{m+1} - x_m$ for SI_r, we have

$$R_i - x_m \le D_i - x_{m+1} - T_i.$$

If $D_i - x_{m+1} > T_i$, then packet P_i can be postponed for transmission at or after x_{m+1} without violating its deadline, and no provision is needed in SI_r. If $D_i - x_{m+1} \le T_i$, then $R_i - x_m \le 0$. According to fact (ii), no over-provision is needed, i.e., $b = 0$. □

Based on Lemma 1, we can readily derive the minimum SP_r required for a node N_r in the worst case. Since the length of the active window (R_i, D_i) impacts the overprovisioning amount and hence the SP_r value, we will consider two possible cases: (i) $(D_i - R_i) \ge 2 * T_i$, and (ii) $(D_i - R_i) < 2 * T_i$. We describe our findings in the following theorem.

THEOREM 1. *Given task τ_i, if $D_i \ge R_i + 2T_i$, the SP_r required to be provisioned for N_r in order to guarantee the transmission of P_i before its deadline D_i is determined as*

$$SP_r = \begin{cases} SI_r \text{ - } (D_i \text{ - } R_i) + 2T_i & \text{if } SI_r > (D_i \text{ - } R_i) \text{ - } T_i \\ T_i & \text{otherwise} \end{cases} \tag{3}$$

On the other hand, if $D_i < R_i + 2T_i$, transmission of P_i by its deadline cannot be guaranteed irrespective of the duration of SP_r for a given SI_r.

Proof: Based on the definition of the over-provision amount, b, as given in Lemma 1, we have $SP_r = b + T_i$. Therefore, from Lemma 1, we immediately obtain (3).

However, (3) only gives a meaningful value when $D_i \ge R_i + 2T_i$. For $D_i < R_i + 2T_i$, if $SI_r > D_i - R_i - T_i$, we have $SP_r > SI_r$ which cannot be satisfied. If $SI_r \le D_i - R_i - T_i$, then $SI_r < T_i$, which makes it impossible to transmit P_i within SI_r. Therefore, if $D_i < R_i + 2T_i$, no provision of SP_r exists that can successfully transmit P_i. □

Theorem 1 suggests how SI_r and SP_r values can be selected to guarantee on-time delivery of real-time packets. As a direct consequence of Theorem 1, we obtain the required bandwidth reservation BW_r at node N_r, given a single traffic-generating task τ_i, as

$$BW_r \ge \begin{cases} 1 - \frac{(D_i - R_i) + 2T_i}{SI_r} & \text{if } SI_r > (D_i - R_i) - T_i \\ T_i / SI_r & \text{otherwise} \end{cases}$$

From Theorem 1, we also see that any $SI_r \le D_i - R_i - T_i$ requires only the smallest provision for SP_r that is equal to the packet transmission time T_i. This leads to the following conclusions for the optimal SI_r and the optimum bandwidth reservation BW_r^*, which are expressed as

$$SI_r^* = D_i - R_i - T_i, \tag{4}$$

$$BW_r^* = \frac{T_i}{D_i - R_i - T_i}. \tag{5}$$

Theorem 1 also leads to two other consequences. First, since packet preemption (or splitting) is not allowed, the active window of a packet must be at least twice the worst-case packet transmission time in order to guarantee feasibility. This is to accommodate the worst-case misalignment between SP_r and the active window (R_i, D_i). Second, it validates the intuitive conclusion that the larger the D_i, the lower the bandwidth requirement.

Figure 3 uses an example to demonstrate the conclusions from this section. It shows the bandwidth requirement BW_r over different SI_r for a task that releases a packet with the worst-case transmission time of 2 time units at the end of its worst-case execution time of 5 (i.e., packet release time is 5). The solid and dotted lines represent the cases when the packet deadline is 35 and 65, respectively. The R_i and D_i values of these packets are relative to the corresponding job release times. BW_r decreases with increasing SI_r and reaches its minimum at $SI_r^* = 28$ and 58, respectively for the two cases. Further increase in SI_r results in a corresponding increase in SP_r as given by Equation 3, which causes BW_r to grow. Also with larger packet deadlines, the optimal SI_r^* is larger and the corresponding SP_r for $SI_r > SI_r^*$ is smaller thereby leading to lower bandwidth requirements.

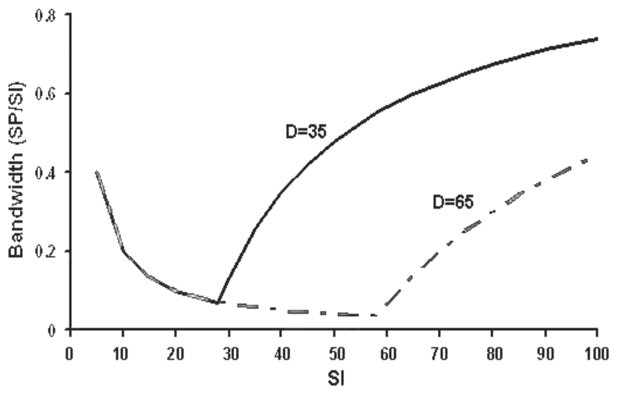


Figure 3: Bandwidth requirements for different SI_r and effects of larger packet deadlines.

Bandwidth Negotiation. From the above conclusions, we propose that a node N_r executing a single traffic-generating task always request an SI_r less than or equal to $(D_i - R_i) - T_i$ and an SP_r equal to T_i. The actual SI_r determined by the BS (based on the requests of all nodes connected to a BS) may differ from the requested SI_r, requiring N_r to recompute and renegotiate SP_r based on Theorem 1.

4. MULTIPLE REAL-TIME STREAMS

Most real-time systems must deal with multiple traffic-generating tasks. This section discusses how the earlier analysis can be extended to multiple traffic-generating tasks at node N_r. We consider a set of periodic tasks $\tau = \{\tau_1, \cdots, \tau_n\}$ that generate a set of packets $\mathcal{P} = \{P_1, \cdots, P_n\}$ in each of their periodic invocations. We first identify the SI_r value that would require the minimal SP_r at a node with multiple traffic-generating tasks. Then we analyze the case when the SI_r provisioned by the BS is greater than the requested value.

4.1 Identification of SI_r that minimizes SP_r

The discussion in Section 3 showed that for a single traffic-generating task, SP_r is minimum, i.e., $SP_r = T_i$, when $SI_r \leq D_i - R_i - T_i$. Therefore, it is natural to first identify when SP_r is minimized for multiple packet-generating tasks. This can be readily done by applying Theorem 1. The conclusion is given in the following theorem.

Theorem 2. *Given a set of packets $\mathcal{P}$ at node N_r, if $SI_r \leq \min_{\tau_i \in \tau}\{(D_i^k - R_i^k) - T_i\}$, then the required SP_r is the minimum, i.e., $SP_r = \sum_{P_i \in \mathcal{P}} T_i$.*

Proof: For any packet P_j, since $D_j - R_j - T_j \geq \min_{P_i \in \mathcal{P}}\{(D_i^k - R_i^k) - T_i\} \geq SI_r$, by regrouping the terms and substituting $x_{m+1} - x_m$ for SI_r, we obtain

$$R_j - x_m \leq D_j - x_{m+1} - T_j.$$

Similar to the arguments used in proving Case 2 in Lemma 1, we have $R_j \leq x_m$ as long as the corresponding packet is to be transmitted in SI_r. In the worst case, each packet is released at or before the start of SP_r. Therefore, to transmit all packets, the minimum SP_r required is $\sum_{\tau_i \in \tau} T_i$. □

Using Equation 4, the optimal SI_r for node N_r with multiple packet-generating tasks that requires minimal SP_r is

$$SI_r^* = min(SI_{r,i}^*) \;\forall\; P_i \in \mathcal{P}. \quad (6)$$

Thus based on Theorem 2, we propose that N_r always request SI_r^* from the BS in order to achieve minimal bandwidth allocation in the network. However, such SI request may not always be satisfied. The next subsection considers this scenario.

4.2 Computation of SP_r when $SI_r > SI_r^*$

As described earlier, it may not be possible to always provision SI_r^* to every node if the BS experiences heavy traffic load. So we consider and analyze the case when the provisioned SI_r is greater than SI_r^*. In such a case, the SP_r for certain (or all) packets are required to be greater than the packet transmission time. We now define and examine the worst-case scenario that requires the largest SP_r.

The worst-case scenario identifies the worst-case phase shifts between the release times of the generated packets and the start of the SI_r. This is important because the phase-shifts between packet releases and the start of SI_r, as we have shown in the single packet-generating task case, are entirely responsible for the overprovisioning amount required to account for those packets released after the start of an SI_r. The following lemma identifies and constructs the worst-case phase shifts between packet releases to compute the required SP_r.

Lemma 2. *Given $SI_r > SI_r^*$, let x_m be the starting time of the m^{th} invocation of SP_r. The worst case that leads to the maximum required SP_r in $[x_m, x_{m+1}]$ occurs when the release time of every packet $P_i \in \mathcal{P}$ causes its deadline to satisfy $D_i = x_{m+1} + T_i - \Delta$, where $\Delta \geq 0$ is the smallest time granularity supported.*

Proof: We first construct the worst case as specified in the lemma and show that any deviation from the case only results in an SP_r that is smaller than or equal to that of the worst case.

We make use of the assumption that $SI_r \leq min\{p_i\}$ (see the end of Section 2) and the fact that one packet is generated during each task period of every packet-generating task. Then it directly follows that at most one packet from each task needs to be transmitted during $[x_m, x_{m+1}]$. To construct the worst case, we assume that each task has exactly one packet to be transmitted in $[x_m, x_{m+1}]$.

The set of n packets in $\mathcal{P}$ is classified into two sub-sets: the set of packets that satisfy $D_i - R_i - T_i < SI_r$ which we denote as $\mathcal{P}_s$, and the rest as $\mathcal{P}_g$. Let j $(0 < j \leq n)$ denote the number of packets in $\mathcal{P}_s$. An example of the worst-case is illustrated in Figure 4, where the timeline is shown in the middle, packets in $\mathcal{P}_s$ are shown above the timeline, and packets in $\mathcal{P}_g$ are shown below the timeline.

Now we show that a violation of the condition stated in Lemma 2 will only result in an SP_r that is smaller than or equal to that of the worst case. The condition can be violated by packets either in $\mathcal{P}_s$ or $\mathcal{P}_g$:

- **Case 1**: consider a packet P_k in set $\mathcal{P}_g$ that violates this condition. In this case, its active window (R_k, D_k) is shifted either to the left or to the right of x_{m+1}. If it is shifted to the left (i.e., $D_k - x_{m+1} < T_k - \Delta$), then P_k can be transmitted in the previous invocation of

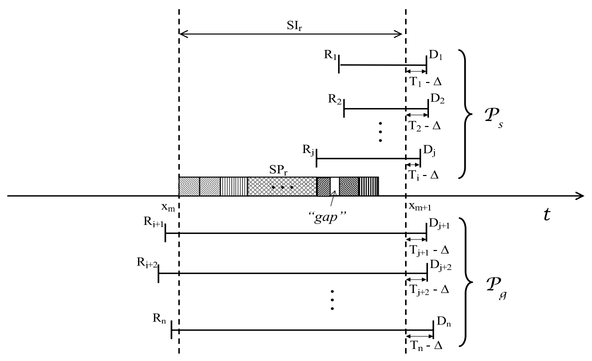


Figure 4: Illustration of an example of the worst case described in Lemma 2. The packets in $\mathcal{P}$ are classified into two sub-sets: the set of packets satisfying $D_i - R_i - T_i < SI_r$ (denoted as $\mathcal{P}_s$), and the rest (denoted as $\mathcal{P}_g$).

SP_r. On the other hand, if (R_k, D_k) is shifted to the right (i.e., $D_k - x_{m+1} > T_k - \Delta$), then transmission of P_k can be delayed to the next invocation of SP_r without violating D_k. Thus violation of the condition in this case only reduces the required SP_r.

- **Case 2**: let packet P_l in $\mathcal{P}_s$ violate the condition. Similar to the previous case, its active window is shifted either to the left or right of x_{m+1}. If it is shifted to the right, this can be analyzed similar to Case 1 and P_l can be transmitted in the next invocation of SP_r. As a result, the required SP_r is lower in this case. The scenario when (R_l, D_l) is shifted to the left requires careful consideration. If the length of this shift is less than SI_r, then P_l is required to be transmitted in the current SP_r. However, any shift to the left will only reduce the required SP_r since the overprovisioning amount B described in Lemma 1 is lowered in such a case.

Thus the violation of any of the two conditions only lowers the required SP_r. Hence it is proved that the worst-case SP_r corresponds to the above identified scenario. □

The importance of Lemma 2 is that it defines precisely the worst-case phase shifts of the packet active windows. Given these phase shifts, the SP_r amount needed to transmit all the packets can be computed. We now propose a mechanism to compute the required SP_r for multiple packet-generating tasks using the identified worst-case scenario. The active windows of the packets generated by the given tasks are aligned with respect to an SI_r invocation in such a way that the condition in Lemma 2 is satisfied for all packets. To compute the SP_r provision, the aligned windows of the packets need to be scanned to consider the "overlaps" (which lower the overprovisioning amount) and "gaps" (which increase the overprovisioning amount) between the release and transmission times of consecutively aligned packets (see Figure 4).

Algorithm 1 describes the details of a linear-time algorithm for calculating the minimal SP_r required at node N_r in the worst-case. The required SP_r is computed by scanning across the active windows of the generated packets and determining the portion of SP_r provision required in each window. In the algorithm, t_{sp}, initialized with the starting time of SI_r, is adjusted incrementally to mark the accumulated sum of the required SP_r portion for each packet. During the scanning process, if the start of a packet's active window overlaps with the duration of the t_{sp} computed thus far, the value of the t_{sp} duration is increased by the transmission time of the packet (line 12 in Algorithm 1). Note that such a scenario occurs when a packet is released before the end of the currently computed t_{sp} window. In the absence of any overlaps, the duration of t_{sp} is extended until the release time of the considered packet and further increased by the time required for its transmission (line 14 in Algorithm 1). This ensures that the gap that exists between the end of the previous computed t_{sp} duration and the release time of the scanned packet is considered. The duration of t_{sp} at the end of the scan of all generated packets is then assigned as the SP_r required to be provisioned to N_r.

Algorithm 1 Computing minimal SP_r for the worst-case

```
Require: (i) Set of n generated packets sorted in the
    increasing order of release-times. (ii) The $SI_r$ ($> SI_r^*$)
    provisioned by the BS, where $\{x_m, x_{m+1}\}$ denote the
    start and end of an $SI_r$ invocation.

 1: worst-case_construct()
 2: compute_$SP_r$()

 3: worst-case_construct():
 4:   for packet i = 1 to n
 5:     align packet $i$ such that $D_i - x_{m+1} = T_i - \Delta$
 6:   end for

 7: compute_SP_r():
 8:   $t_{start} = x_m$
 9:   $t_{sp} = t_{start}$
10:   for packet i = 1 to n
11:     if ($t_{sp} \geq R_i$)
12:       $t_{sp} += T_i$
13:     else
14:       $t_{sp} = R_i + T_i$
15:     end if
16:   end for
17:   $SP_r = t_{sp} - t_{start}$
```

The correctness and time complexity of Algorithm 1 are given in the following theorem.

THEOREM 3. *Given $SI_r > min\{(D_i - R_i) - T_i, \forall P_i \in \mathcal{P}\}$, Algorithm 1 finds the optimal SP_r required in the worst case in $O(n)$ time where n is the total number of tasks in τ.*

Proof: Since Algorithm 1 scans each of the n generated packets exactly once, its computational complexity is $O(n)$.

We prove that Algorithm 1 always finds the optimal SP_r required in the worst case by considering two situations:

- In the event that the transmission times of all generated packets overlap in the SI_r under consideration (i.e., line 14 is never executed in the algorithm), then the calculated SP_r will be equal to $\sum_{i=1}^{n} T_i$. Since the required SP_r cannot be lower than this (from Theorem 2), Algorithm 1 gives the optimum SP_r required.

- In the absence of any overlaps between the packet transmission times, the "gaps" that exist between them need to be considered in computing the required SP_r. In this case, we show that it is impossible to avoid including these gaps in the required SP_r. This is because: (i) it is evident that the packets whose release times R_i are later than the occurrence of this "gap" cannot be transmitted in the duration of this gap since they have not been released yet; (ii) the packets with release time R_i earlier than the occurrence of this "gap" also cannot be transmitted in this duration since this only shifts the "gap" to an earlier interval in time (i.e, to the time interval in which this packet is actually being transmitted). Thus in both cases, Algorithm 1 gives the optimal SP_r.

Hence this theorem is proved. $\square$

So far, we have shown the computation of SI_r and SP_r for the traffic generated from multiple tasks. Using the conclusions from this analysis, we study the effects of packet deadlines on the reservation parameters and propose guidelines for their assignment. We also use it to devise a scheme for the negotiation of the SP_r and SI_r parameters at each node.

4.3 Deadline selection and bandwidth negotiation

This section describes guidelines for the assignment of packet deadlines and the bandwidth negotiation phase. We will discuss deadline assignment first since it is used in the bandwidth negotiation process.

Guidelines for Packet Deadline Assignment. For multiple traffic-generating tasks, the deadlines of all generated packets do not have a uniform effect on the resource requirements. This can be inferred from Theorem 2 where the optimal SI_r^* is determined only by the packet with the smallest $D_i - R_i - T_i$. Thus any increase in the deadlines of the other packets does not lead to larger values for the optimal SI_r^*.

On the other hand, the SP_r requirement for any $SI_r > SI_r^*$ is heavily dependent on the packets whose $D_i - R_i - T_i$ is less than SI_r (i.e, packets identified in set $\mathcal{P}_s$ in Lemma 2). This is because the determination of the required SP_r in Algorithm 1 is dominated by t_{sp} computed in line 14. The duration of t_{sp} in line 14 is extended to cover the release of packets that occur after the start of the SI_r considered in the worst-case scenario. This case concerns the packets that have $D_i - R_i - T_i < SI_r$ and are classified as set $\mathcal{P}_s$. Therefore to lower the value of t_{sp} computed in this case, the deadline of these packets (or their release times if control over the task scheduling mechanism and the task execution speeds are available) need to be relaxed. However, note that t_{sp} is simply computed as the sum of the transmission times for packets that are released before the start of the considered SI_r i.e, packets in set $\mathcal{P}_g$. Thus increasing the deadlines of packets in $\mathcal{P}_g$ will not result in a reduction of the required SP_r. This also implies that increasing the deadlines of the packets in $\mathcal{P}_s$ beyond $SI_r + R_i + T_i$ will not lower SP_r. *Hence contrary to common perception, it is found that arbitrarily increasing the deadline of any generated packet does not always lower the resource requirements at a node.*

Based on these conclusions, the following packet deadline assignment guidelines are proposed (when flexibility in their assignment is available):

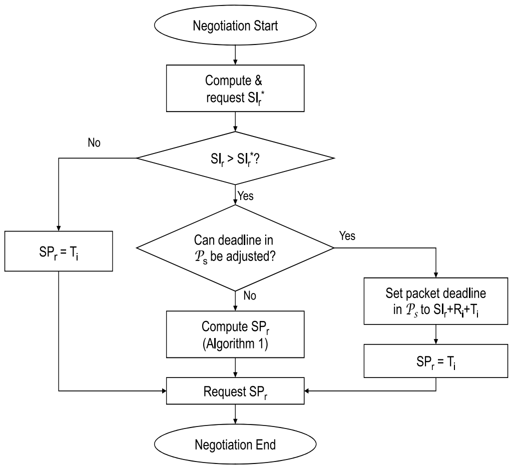

Negotiation Start
Compute & request SI_r^*
No
$SI_r > SI_r^*$?
Yes
$SP_r = T_i$
Can deadline in $\mathcal{P}_s$ be adjusted?
Yes
No
Set packet deadline in $\mathcal{P}_s$ to $SI_r+R_i+T_i$
Compute SP_r (Algorithm 1)
$SP_r = T_i$
Request SP_r
Negotiation End

Figure 5: Bandwidth negotiation scheme for multiple traffic-generating tasks. The SI_r is allocated by the BS.

- the deadlines of packets with the smallest $D_i - R_i - T_i$ be increased so that the optimal SI_r is larger;
- for the case $SI_r > SI_r^*$, the deadlines of the packets that satisfy $D_i - R_i - T_i < SI_r$ be adjusted to be close to $SI_r + R_i + T_i$ so that the required SP_r is lowered.

Bandwidth Negotiation. The bandwidth negotiation is performed similar to the single-packet generating task case described at the end of Section 3. Figure 5 describes the steps involved in bandwidth negotiation by a node N_r with multiple packet-generating tasks. In this scheme, N_r initially requests the minimum of the $SI_{r,i}^*$ computed for all individual packet-generating tasks since it requires the smallest SP_r provision. However, if the BS indicates that the SI_r it can provision is greater than SI_r^*, N_r is required to do either of the following: (i) relax the deadline constraints of the packets in set $\mathcal{P}_s$ to $SI_r + R_i + T_i$, so that only the minimal SP_r (from Theorem 2) is still required, and (ii) in the absence of flexibility in adapting packet deadlines, use Algorithm 1 to compute and request the SP_r required for the given SI_r considering the worst-case scenario described in Lemma 2. Our future work will address the scenario when the SP_r provisioned by the BS is smaller than the requested value.

4.4 Extension of analysis

This section presents a discussion of possible extensions of our formulations of the reservation parameters to relax earlier assumptions and cover more general scenarios.

Incoming Traffic. Our analysis can be extended to consider incoming traffic at the nodes by modeling the BS as a transmitting node. This is possible because the BS is responsible for forwarding the packets received from the connected nodes to their corresponding destination nodes. Thus the proposed formulations can be extended to this scenario by considering incoming traffic as an additional real-time stream, thereby computing SP and SI parameters for the traffic coming from the BS.

Task	Period (ms)	Worst-case Execution Time (ms)	Packet Deadline (ms)	Worst-case Packet Transmission Time (ms)
τ_1	300	60	400	20
τ_2	400	100	525	5
τ_3	450	60	565	5
τ_4	250	50	450	10

Table 1: Description of the task and packet parameters used in the experiments.

Multiple Packet Generations per Job Execution. Our analysis also applies to a traffic model where multiple packets are generated in each job execution. This is because each distinct packet release in a job invocation can be modeled as a single packet generated by an individual task. Therefore the formulations for multiple traffic-generating tasks presented in this section can be applied to this case.

Network Dynamics. In real world applications, wireless nodes join and leave a network dynamically and traffic loads also vary over time. Each individual node may need to recalculate and resubmit its resource requirements dynamically based on the interactions and negotiations with the BS as well as the change of its own resource requirements. In such situations, the proposed strategy could work together closely with the QoS scheduling approaches at the BS side, such as [5, 9, 13], to achieve the QoS guarantees and efficient resource utilization across the entire network. Due to the low computation cost of the proposed strategy, such a dynamic procedure will not likely yield significant burden at each node.

5. EXPERIMENTAL EVALUATIONS

This section describes the setup and results from our evaluations of the presented mechanisms and guidelines.

5.1 Simulation setup

The mechanisms were evaluated using an event-driven simulator built in Java. Each node is simulated to execute our proposed mechanisms in computing and negotiating the required SI_r and SP_r values based on the packet parameters.

The evaluations presented in this section were obtained with the taskset and packet parameters shown in Table 1. The taskset contains four traffic-generating tasks, which can be considered as, for example, different sensing functions that generate real-time streams. An EDF-based task scheduling algorithm was employed and the deadline d_i for each job J_i^k was set to the end of their respective periods. A packet was generated at the completion of each job (which represents the worst case). Since all jobs satisfy the schedulability requirement for EDF scheduling (i.e., utilization $\leq 100\%$), a job always completes execution by its deadline. Thus a job J_i^k releases a packet before or at its deadline d_i, which in the worst case gives $R_i^k = d_i$. The experiments were run over a duration of 20 times the least-common-multiple of the task periods employed.

Task periods and packet deadlines are generally given by the corresponding applications. A great deal of work has been done on estimating task execution time. Below we briefly discuss the derivation of the worst case on the packet transmission time. Wireless channel conditions are time-varying and error-prone. Most wireless networks support multiple transmission rates and rely on rate-adaptation algorithms to choose the optimal rates that matches the instant channel conditions. Retransmissions are usually allowed if a transmission fails, until a retry limit is reached. Here, we derive the equations for the worst-case transmission time following the retransmission policy defined for the HCCA mode in 802.11e, in which a node can start a retransmission after a time period of PIFS (PCF InterFrame Space) if no ACK is received from the BS. We have the following formula for the worst-case transmission time:

$$\begin{aligned} T_{WS} = (t(L_{data}, M_{min}) + PIFS) * RetryLimit \\ -PIFS + SIFS + t(L_{ack}, M_{ack}), \end{aligned} \quad (7)$$

where L_{data} is the length (in bytes) of the data frame (including the MAC header), L_{ack} is the length (in bytes) of the ACK frame, M_{min} is the transmission mode that supports the lowest transmission rate, and M_{ack} is the transmission mode used by the ACK frame. *RetryLimit* defines the limit on the number of attempts to transmit each frame[1], after which the frame should be discarded. The value of SIFS (Short InterFrame Space), PIFS, and the expression for function $t(\ell, m)$ depend on the modulation schemes used in the physical (PHY) layer. For example, for 802.11b, the SIFS and PIFS are $10\mu s$ and $30\mu s$ respectively, and

$$\begin{aligned} t(\ell, m) &= t_{PLCP\_Preamble} + t_{PLCP\_Header} + 8 * \ell / r(m) \\ &= 192\mu s + 8 * \ell / r(m), \end{aligned} \quad (8)$$

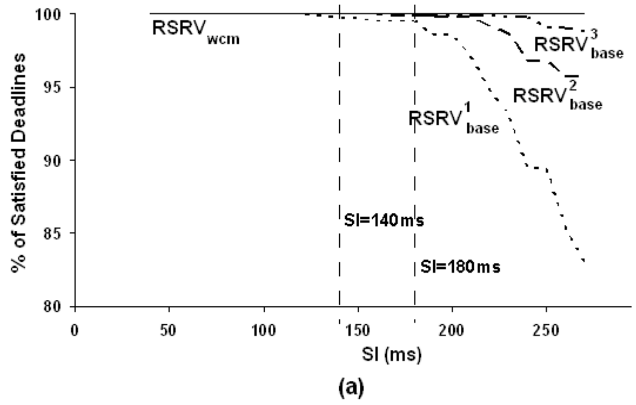
where $t_{PLCP\_Preamble}$ and t_{PLCP_Header} are the times used to transmit the Preamble and Header components of the PLCP (Physical Layer Convergence Procedure) sublayer respectively; $r(m)$ is the data rate supported by transmission mode m. The formula of $t(\ell, m)$ for 802.11a and 802.11g physical layers could be derived similarly.

The worst-case transmission times used in the experiments were derived based on the above analysis. It is worthwhile noting that since the lowest transmission rate was used for each transmission attempt in the analysis, the calculated worst-case transmission time could be much larger than the actual case transmission time. A less conservative strategy is to periodically recalculate the worst-case transmission time dynamically based on the feedback from the PHY layer, i.e., the statistics of the recent channel conditions, the effective transmission rate, and the packet size distribution, etc. This may lead to more efficient utilization of channel resources, however, less confidence in the QoS guarantee.

5.2 Simulation results

We now illustrate the performance of our strategies in satisfying the timeliness constraints of the traffic and the effectiveness of the guidelines for packet deadline assignment.

[1]In 802.11, retransmissions of short frames (length $\leq$ RTS Threshold) and long frames (length $>$ RTS Threshold) are treated separately using two different limits on the number of attempts, namely, ShortRetryLimit and LongRetryLimit. For ease of explanation, we use a single RetryLimit.



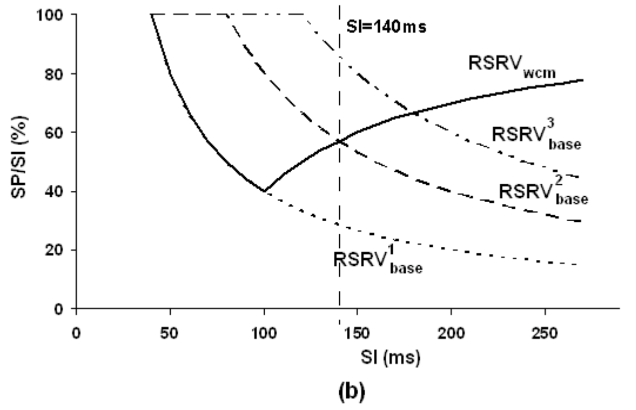


Figure 6: (a) Percentage of satisfied deadlines and (b) bandwidth requirements over different SI_r, for the baseline mechanisms ($RSRV^c_{base}$) and our proposed approach ($RSRV_{wcm}$).

Performance of Resource Reservation Mechanisms. The performance of our proposed approach that computes the minimal resources required in the worst case for a given traffic-generating taskset is evaluated against baseline cases. We use $RSRV_{wcm}$ to denote our approach for computing the 'worst-case minimum' SP_r values. The baseline cases are identified as the mechanisms that reserve SP_r for any given SI_r as $c * \sum T_i$ where 'c' is some pre-selected constant. These mechanisms are represented as $RSRV^c_{base}$ in our evaluations and we consider the cases when 'c' is 1, 2, and 3, which results in SP_r provisions of 40ms, 80ms, and 120ms respectively for the taskset in Table 1. Figure 6 compares the number of satisfied packet deadlines and the bandwidth reservations (SP_r/SI_r) at a node between $RSRV_{wcm}$ and the different baseline cases. There are two observations of interest in these comparisons which are described below.

First, from Figure 6(a) we observe that packet deadline violations (i.e., satisfied deadlines < 100%) for the baseline cases increase as SI_r increases. This is because the provisioned SP_r does not include the overprovisioning amount required to cover the "gaps" between packet releases and the start of SI_r. Our approach satisfies all the deadlines for a given SI_r with the SP_r computed from Algorithm 1 since it considers the worst-case phase-shifts between packet releases and the start of SI_r as defined in Lemma 2.

Second, it can be observed that our approach satisfies packet deadlines without overprovisioning resources by carefully considering the SI_r values at which the $RSRV_{wcm}$ curve intersects with the different $RSRV^c_{base}$ curves in Figure 6(b). We observe from Figure 6(a) that the $RSRV_{wcm}$ and $RSRV^c_{base}$ mechanisms satisfy all deadlines for SI_r values smaller than the value at their intersection in Figure 6(b). However it is important to note that $RSRV_{wcm}$ reserves lower bandwidth than the baseline cases ($RSRV_{wcm}$ makes similar reservations as $RSRV^1_{base}$) for SI_r in this range. Thus our approach performs better in satisfying deadlines with minimal bandwidth reservations for this range. On the other hand, for SI_r greater than the values at the intersections, $RSRV_{wcm}$ reserves higher bandwidth than the baseline cases. But observe that $RSRV_{wcm}$ satisfies all deadlines while deadline violations occur in the $RSRV^c_{base}$ mechanisms. As an example, consider $RSRV_{wcm}$ and $RSRV^2_{base}$ which intersect at $SI_r = 140$ms in Figure 6(b). We observe that $RSRV_{wcm}$ has significantly lower bandwidth reservations compared to $RSRV^2_{base}$ for $SI_r < 140$ms and that it satisfies all deadlines. Deadlines are missed in $RSRV^2_{base}$ for $SI_r > 140$ms while $RSRV_{wcm}$ satisfies all deadlines by computing the worst-case SP_r value using Algorithm 1.

Taskset	Task & Packet Parameters
TS_a	Same as in Table 1
TS_b	Same as in Table 1 except $D_1 = 430$ms
TS_c	Same as in Table 1 except $D_1 = 500$ms, $D_2 = 585$ms, $D_3 = 635$ms
TS_d	Same as in Table 1 except $D_4 = 480$ms

Table 2: Tasksets used in Figure 7.

Packet Deadline Effects. To study how task deadline adjustments affect packet scheduling, we vary the deadlines for the tasksets in Table 1 and present the modified tasksets in Table 2. Figure 7 compares the number of packet deadline violations and bandwidth requirements for the four tasksets in Table 2. The SP_r value for each case in Figure 7 is set to the sum of the packet transmission times which is 40ms here (i.e., corresponding to $RSRV^1_{base}$). It is observed that when the deadline for the packet that has the minimum $D_i - R_i - T_i$ is enlarged (i.e., the deadline for the packet generated by τ_1 in TS_a is increased to 430ms), the bandwidth requirements are lowered since the optimal SI^*_r increases. This effect can be observed similarly for the case when the deadlines for packets generated by τ_1, τ_2 and τ_3 in TS_a are increased such that the minimum of all $D_i - R_i - T_i$ is higher. On the other hand, any increase in the deadline for the packets released by other tasks such as τ_4 do not result in any reduction of bandwidth requirements or number of deadline violations. This is seen in Figure 7 where the curve representing TS_d overlaps the curve for TS_a in terms of bandwidth requirements as well as the deadline violations.

Summary. The above evaluations show that our proposed reservation strategies satisfy the requirements of real-time traffic with economical resource reservation (Figure 6). Our conclusion that increasing the deadlines of any random packet does not always lower the resource requirements is also shown to hold true (Figure 7).

The performance of our approach for the case when SI_r is greater than the optimal SI^*_r, where either the deadlines for packets with $D_i - R_i - T_i < SI_r$ are increased or SP_r is computed using Algorithm 1, can be verified from Figures 6 & 7. As an example, consider the taskset TS_a in Table 2

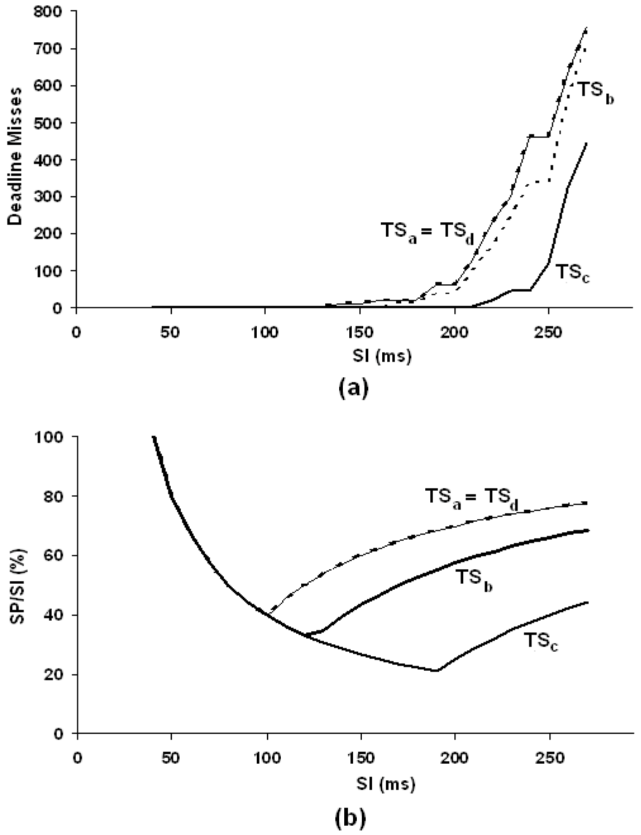


Figure 7: (a) Number of deadline violations and (b) bandwidth requirements for tasksets with different packet deadlines.

for which SI_r^* is 80ms. Assume that the provisioned SI_r is 180ms. In this case, deadlines are violated with the smallest SP_r provision of 40ms which is observed in Figure 6(a) (corresponding to $RSRV_{base}^1$). When the deadlines of packets that satisfy $D_i - R_i - T_i < 180$ms (i.e., packets generated by τ_1, τ_2 and τ_3) are increased to $SI_r + R_i + T_i$, all deadlines are satisfied at $SI_r = 180$ms with an SP_r of 40ms (observed from the curve for taskset TS_c in Figure 7(a)). In the absence of any flexibility in changing these deadlines, the SP_r computed from Algorithm 1 for the given SI_r satisfies all deadlines as seen earlier.

6. CONCLUSIONS

In this work, we analyzed the worst-case scenarios for reservation-based wireless real-time traffic and used it to derive formulae for determining the minimal values for the negotiable parameters used in provisioning channel accesses in wireless embedded environments. The proposed approach to compute the reservation parameters satisfies the timeliness requirements of the generated traffic without overprovisioning resources. This work also investigated the assignment of packet transmission deadlines and their impact on the resource requirements. Based on this study, guidelines for the assignment of deadlines were presented.

7. REFERENCES

[1] M. Adamou, S. Khanna, I. Lee, I. Shin, and S. Zhou. Fair real-time traffic scheduling over a wireless LAN. In *Proc. of IEEE RTSS'01*, December 2001.

[2] P. Ansel, Q. Ni, and T. Turletti. An efficient scheduling scheme for IEEE 802.11e. In *Proc. of WiOpt'04*, March 2004.

[3] G. Boggia, P. Camarda, L. A. Grieco, and S. Mascolo. Feedback-based control for providing real-time services with the 802.11e MAC. *IEEE/ACM Transactions on Networking*, 15(2):323–333, 2007.

[4] D. Bruneo, L. Paladina, M. Paone, and A. Puliafito. Resource reservation in mobile wireless networks. In *Proc. of IEEE ISCC'04*, 2004.

[5] C. Cicconetti, L. Lenzini, E. Mingozzi, and G. Stea. Design and performance analysis of the real-time HCCA scheduler for IEEE 802.11e WLANs. *Computer Networks*, 51(9):2311–2325, 2007.

[6] C. Cicconetti, L. Lenzini, E. Mingozzi, and G. Stea. An efficient cross layer scheduler for multimedia traffic in wireless local area networks with IEEE 802.11e HCCA. *ACM Mobile Computing and Communication Review*, 11(3):31–46, 2007.

[7] A. Easwaran, M. Anand, and I. Lee. Compositional analysis framework using EDP resource models. In *Proc. of IEEE RTSS'07*, December 2007.

[8] T. Facchinetti, L. Almeida, G. Buttazzo, and C. Marchini. Real-time resource reservation protocol for wireless mobile ad hoc networks. In *Proc. of IEEE RTSS'04*, 2004.

[9] A. Grilo, M. Macedo, and M. Nunes. A scheduling algorithm for QoS support in IEEE 802.11e networks. *IEEE Wireless Communications*, 10(3):36–43, 2003.

[10] *IEEE 802.11e Standard, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Medium Access Control (MAC) Enhancements for Quality of Service (QoS).*

[11] P. Littieri and M. B. Srivastava. Adpative frame length control for improving wireless link throughput, range and energy efficiency. In *Proc. of IEEE INFOCOM'98*, March 1998.

[12] S. Mangold, S. Choi, P. May, O. Klein, G. Hiertz, and L. Stibor. IEEE 802.11e wireless LAN for Quality of Service. In *Proc. of European Wireless (EW)*, February 2002.

[13] M. M. Rashid, E. Hossain, and V. K. Bhargava. HCCA scheduler design for guaranteed QoS in IEEE 802.11e based WLANs. In *Proc. of IEEE WCNC'07*, pages 1538–1543, 2007.

[14] M. V. D. Schaar, Y. Andreopoulos, and Z. Hu. Optimized scalable video streaming over IEEE 802.11a/e HCCA wireless networks under delay constraints. *IEEE Transactions on Mobile Computing*, 5(6):755–768, 2006.