Admission Control and Scheduling of Isochronous and Asynchronous Traffic in IEEE 802.11ad MAC

Anirudha Sahoo[1], Pu Tian[2,3], Tanguy Ropitault[3,4], Steve Blandino[3,4] and Nada Golmie[1]
[1]National Institute of Standards and Technology, Gaithersburg, Maryland, USA
[2]Towson University, Towson, Maryland, USA
[3]Associate, National Institute of Standards and Technology, Gaithersburg, Maryland, USA
[4]Prometheus Computing LLC, Bethesda, Maryland, USA
Email: {anirudha.sahoo, pu.tian, tanguy.ropitault, steve.blandino, nada.golmie}@nist.gov

***Abstract*—The next generation WiFi such as IEEE 802.11ad and 802.11ay can provide stringent Quality of Service (QoS) due to its support of contention free channel access called *Service Period*. IEEE 802.11ad supports two types of user traffic: *isochronous* and *asynchronous*. These user traffic need guaranteed Service Period duration before their periods. Hence, admission control plays an important role in an IEEE 802.11ad system. In an earlier work we studied admission control only for isochronous requests. In this paper, we present admission control and scheduling algorithms which can handle both types of requests. We devise a proportional fair and linear run time complexity algorithm that treats asynchronous requests as periodic requests, because of which it overallocates resources to the asynchronous requests. The conditions of possible performance loss due to this overallocation are analyzed. We provide detailed simulation results which show that presence of asynchronous request degrades performance of isochronous requests in terms of number of admitted requests and channel utilization. But, the smaller number of admitted isochronous requests perform better in terms of channel allocation time and delay.**

***Index Terms*—admission control, scheduling, IEEE 802.11ad, MAC, isochronous, asynchronous**

I. Introduction

Availability of a large amount of unlicensed bandwidth in the millimeter wave (mmWave) 60 GHz band and proliferation of high data rate and low latency applications, such as Virtual Reality, 8K TV, have led to development of standards for the next generation WiFi such as IEEE 802.11ad and its successor 802.11ay. IEEE 802.11ad Medium Access Control (MAC) has two types of user traffic: *isochronous* and *asynchronous*. Isochronous traffic is a periodic traffic which must be allocated certain minimum *service period* (SP) duration in every period. An SP is allocated to a pair of stations to have a guaranteed contention free communication between them. Asynchronous traffic, on the other hand, is a one time request. One of the ways asynchronous traffic is requested is using *Add Traffic Stream* (ADDTS) request for channel time in subsequent scheduling periods [1]. Hence, the admitted asynchronous request must be guaranteed its requested SP duration before its deadline. In [2], we presented admission control and scheduling algorithms for only isochronous traffic in an IEEE 802.11ad system. However, a real IEEE 802.11ad system will have both kinds of traffic. In this paper, we present admission control and scheduling algorithms which can handle both isochronous and asynchronous requests and hence, is more suitable for a real IEEE 802.11ad system.

Our admission control algorithm, called Admission Control for Isochronous and Asynchronous Requests (ACIAR) deals with asynchronous requests as if they are periodic requests. This enables us to apply admission criterion used in the *Earliest Deadline First* (EDF) scheduling for periodic Central Processing Unit (CPU) tasks in a realtime system [3] and then use EDF to schedule the requests. However, the ACIAR algorithm overallocates resources to asynchronous requests since it allocates resources for multiple periods of the asynchronous requests. Hence, it may lead to performance loss. We provide a detailed analysis of the conditions when possible performance loss may occur due to the overallocation. However, our analysis as well as simulation results show that if the load of asynchronous requests is low, then the performance loss occurs only at high load of isochronous requests. Apart from our earlier work in [2], as far as we know, the only other experimental study of admission control and scheduling for mmWave WiFi system (e.g., IEEE 802.11ad) was reported in [4]. In that work, the authors present a max-min fair scheduling algorithm, but the algorithm does not handle asynchronous requests and has a cubic run time complexity in terms of number of requests. In contrast ACIAR, while being proportionally fair, can handle both isochronous and asynchronous requests and has a linear run time complexity in terms of number of requests. To the best our knowledge, ACIAR is the first admission control algorithm for IEEE 802.11ad that can handle both isochronous and asynchronous requests.

We present detailed performance results of ACIAR algorithm with respect to different performance metrics. Especially, we investigate the impact of asynchronous traffic on the performance of isochronous traffic. Our results show that the presence of asynchronous requests cause the system to admit lesser number of isochronous requests and the admitted isochronous requests achieve lower channel utilization. However, the performance of those admitted isochronous requests in terms of other metrics such as amount of channel time allocated to each request and delay is better when they share the system with asynchronous requests. Our simulation results on overallocation show that the ACIAR algorithm may incur

performance loss when the system runs with a very high isochronous request load.

II. Admission Control and Scheduling of Isochronous and Asynchronous Traffic

A. IEEE 802.11ad Medium Access

The medium access duration in IEEE 802.11ad is divided into an infinite sequence of time intervals, called *Beacon Interval* (BI). The length of a BI duration is specified in Time Units (TU), where $1\,\text{TU} = 1024\,\mu\text{s}$. A BI has two parts: a Beacon Header Interval (BHI) followed by a Data Transmission Interval (DTI). The DTI is primarily used for data transmission among IEEE 802.11ad Stations (STAs) and Personal Basic Service Set (PBSS) Control Point/Access Point (PCP/AP). An IEEE 802.11ad station can access the channel in a DTI using Contention Based Access Period (CBAP) or Service Period (SP) mechanism. When using CBAP, a station uses a contention based scheme called Enhanced Distributed Channel Access (EDCA) [1]. An SP, on the other hand, is used between two stations or between a station and its PCP/AP to have a contention free channel access. Before a DTI period starts, the schedules of CBAP and SP in a DTI are broadcast by the PCP/AP to its stations in a Directional Multi Gigabit (DMG) Beacon frame in the Beacon Transmission Interval (BTI) or in the Announce frame in the Announcement Transmission Interval (ATI) of the BHI [1].

B. IEEE802.11ad Traffic Parameters

The Traffic Specification (TSpec) element in the ADDTS request carries the traffic parameters for which resources need to be allocated. For isochronous requests, channel time allocation duration is repeated in every period, whereas for asynchronous traffic, allocation is one time. The main traffic parameters (used in the DMG TSpec frame) of isochronous traffic are [1], [2]:

- Allocation Period (P): Period over which allocation repeats. It can be an integer multiple or integer fraction of a BI.
- Minimum Allocation (C^{min}): Minimum acceptable allocation in microseconds in each allocation period. If the request is accepted, the PCP/AP must guarantee at least this duration to the STA in every allocation period.
- Maximum Allocation (C^{max}): Requested allocation in microseconds in each allocation period. This is the maximum duration that can be allocated to the user in each allocation period.

Asynchronous requests also use the same DMG TSpec parameters. However, the minimum allocation is one time allocation that must be done before the allocation period (or *deadline*). Maximum allocation field is reserved.

C. Modeling of Isochronous and Asynchronous Traffic

A request T_i is an isochronous (resp. asynchronous) request when $T_i.reqType$ is *ISO* (resp. *ASYN*). The period, minimum allocation, maximum allocation of an isochronous request T_i are denoted by $T_i.P$, $T_i.C^{min}$ and $T_i.C^{max}$ respectively. The actual (or operational) channel time allocation to isochronous request T_i is denoted as $T_i.C^{op}$ and should satisfy $T_i.C^{min} \leq T_i.C^{op} \leq T_i.C^{max}$. The period (or deadline) and minimum allocation of asynchronous request T_i are denoted by $T_i.P$ and $T_i.C^{min}$ respectively. The operational allocation $T_i.C^{op}$, in this case, is equal to $T_i.C^{min}$. An isochronous request has allocation request in every period. Such allocation requests are referred to as *jobs* of the request. Since an asychronous request does not repeat beyond its deadline, it has exactly one job in its lifetime. A job of a request becomes ready to be allocated at the beginning of its period, which is referred to as the *release time* of the job. A job of a request may be allocated one contiguous block or it may be broken into multiple *fragments* to fit smaller idle durations in the BI.

D. CPU Scheduling of Periodic Tasks

CPU scheduling of periodic tasks has been extensively studied in the literature [3], [5]–[7]. Due to periodic property of isochronous traffic, its scheduling can be designed using the theories developed for CPU scheduling of periodic tasks. A periodic task in CPU scheduling has two parameters (C_i, P_i), where C_i is the duration of the task and P_i is the period as well as the deadline of the task. The feasibility or admissibility of a set of n preemptive periodic tasks for an EDF scheduler is given by [3]

$$\sum_{i=1}^{n} \frac{C_i}{P_i} \leq 1. \tag{1}$$

E. Admission Control of Isochronous and Asynchronous Requests

The TSpec of isochronous traffic has a range of duration, unlike the CPU scheduling of a task which has a single duration. So, an admission control algorithm not only determines whether a new request can be admitted or not, but also computes C^{op}. We are able to use the admissibility criteria Eq. (1) by treating asynchronous requests as periodic.

Let us assume that there are $(n-1)$ requests, running at their respective C^{op}s, already in the system. The newly arriving n^{th} request, T_n, which could be an isochronous or asynchronous request, has the operational allocation C_n^{op}. Then T_n is admitted if and only if

$$U + \frac{C_n^{op}}{P_n} \leq 1, \tag{2}$$

where $U = \sum_{i=1}^{(n-1)} \frac{C_i^{op}}{P_i}$ is the utilization of the system due to already admitted requests.

The admissibility criterion specified in Eq. (2) is realized in Algorithm 1 which shows our ACIAR algorithm. If the sum of utilization of existing isochronous requests, asynchronous request and the new request, based on their C^{min}, is more than 1, then the new request is rejected (Line 7). Otherwise, *surplus* utilization over the C^{min}-based utilization is calculated (Line 8). The surplus is distributed to every isochronous request in a proportionally fair manner. Fairness is guaranteed by making sure that $\frac{T_j.C^{op} - T_j.C^{min}}{T_j.C^{max} - T_j.C^{min}}$ is the same for every isochronous requests T_j (Line 15). Note that when $\frac{U_{surplus}}{\Delta u_{tot}}$ is

greater than 1, it implies that there is enough surplus for every isochronous request to be allocated its C^{max}. That is why the term $\min(1, \frac{U_{surplus}}{\Delta u_{tot}})$ is used in Line 15 to prevent $T_j.C^{op}$ going above $T_j.C^{max}$. Noting the *for* loops in Line 12 and in Line 14, each of which iterates over all the isochronous requests (n_{iso} in number) in the system, it is clear that the run time complexity of our ACIAR algorithm is $O(n_{iso})$.

Algorithm 1 Admission Control for Isochronous and Asynchronous Requests (ACIAR)

1: **input:** the new request T_n, the set of existing isochronous requests $\mathcal{S}_{iso}$ and the set of existing asynchronous requests $\mathcal{S}_{async}$
2: **output:** ACCEPT or REJECT; C^{op} of each isochronous request if the new request is accepted.
3: reqType = $T_n.reqType$
4: $u^{min}_{new} = \frac{T_n.C^{min}_n}{T_n.P_n}$
5: $U^{min}_{iso} = \sum_{T_i \in \mathcal{S}_{iso}} \frac{T_i.C^{min}}{T_i.P}$
6: $U^{min}_{async} = \sum_{T_i \in \mathcal{S}_{async}} \frac{T_i.C^{min}}{T_i.P}$
7: **if** $(U^{min}_{iso} + U^{min}_{async} + u^{min}_{new}) > 1$ **then return** REJECT
8: $U_{surplus} = 1 - (U^{min}_{iso} + U^{min}_{async} + u^{min}_{new})$
9: **if** (reqType == ISO) **then** $\mathcal{S}_{iso} = \mathcal{S}_{iso} \cup \{T_n\}$
10: **else** $\mathcal{S}_{async} = \mathcal{S}_{async} \cup \{T_n\}$
11: $\Delta u_{tot} = 0$
12: **for** each $T_j \in \mathcal{S}_{iso}$ **do**
13: $\quad \Delta u_{tot} = \Delta u_{tot} + \frac{T_j.C^{max} - T_j.C^{min}}{T_j.P}$
14: **for** each $T_j \in \mathcal{S}_{iso}$ **do**
15: $\quad T_j.C^{op} = T_j.C^{min} + \min(1, \frac{U_{surplus}}{\Delta u_{tot}}) \cdot (T_j.C^{max} - T_j.C^{min})$
16: **return** ACCEPT

The EDF scheduling of admitted requests can be summed up as follows. The jobs of all the requests in a given BI are sorted in a non-decreasing order of their deadline. The jobs are picked from this sorted list one at a time. A job is allocated its C^{op} duration in the first available space in the BI (which must be after its release time). If the available space is not long enough to accommodate its C^{op}, then the job is fragmented, with the first fragment fully occupying the first available space and the second fragment taken to the next empty space. If the next empty space is long enough, then the second fragment is allocated there. Otherwise, the second fragment is further fragmented into two and the process repeats until the job is fully allocated.

F. Effect of Overallocation to Asynchronous Request

We now discuss the effect of overallocation due to consideration of asynchronous request as periodic. Let U_{async} and U^{min}_{iso} respectively be the minimum utilization of all admitted asynchronous and isochronous requests in the system, i.e.,

$$U_{async} = \sum_{T_i \in \mathcal{S}_{async}} \frac{T_i.C^{min}}{T_i.P}, \tag{3}$$

$$U^{min}_{iso} = \sum_{T_i \in \mathcal{S}_{iso}} \frac{T_i.C^{min}}{T_i.P}. \tag{4}$$

Let $U^{min} = U_{async} + U^{min}_{iso}$ be the total utilization of the system based on minimum duration allocated to isochronous requests. As per the EDF admission control criterion

$$U^{min}_{iso} + U_{async} \leq 1. \tag{5}$$

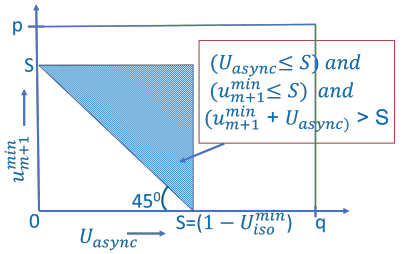


Fig. 1: Analysis of Performance Loss due to Overallocation

Assume that an isochronous request T_{m+1} with traffic parameters $C^{min}_{m+1}, C^{max}_{m+1}, P_{m+1}$ arrives. Let $u^{min}_{m+1} = \frac{C^{min}_{m+1}}{P_{m+1}}$. There are three cases to consider as follows.

$$\text{Case 1}: \quad u^{min}_{m+1} \leq (1 - U^{min}), \tag{6}$$

$$\text{Case 2}: \quad u^{min}_{m+1} > (1 - U^{min}_{iso}), \tag{7}$$

$$\text{Case 3}: (1 - U^{min}) < u^{min}_{m+1} \leq (1 - U^{min}_{iso}). \tag{8}$$

Note that $U^{min} \geq U^{min}_{iso}$ and hence, Case 2 implies $u^{min}_{m+1} > (1 - U^{min})$. When Eq. (6) is satisfied ACIAR algorithm would accept T_{m+1}. When Eq. (7) or Eq. (8) holds, then it would reject the request. However, when Eq. (8) holds, u^{min}_{m+1} is more than available utilization $(1 - U^{min})$ which includes the utilization of asynchronous requests which are considered periodic. Hence, there could be performance loss when Eq. (8) holds.

Using $U^{min} = U_{async} + U^{min}_{iso}$ in Case 3, we have

$$u^{min}_{m+1} > (1 - (U_{async} + U^{min}_{iso})),$$

$$i.e., \; u^{min}_{m+1} + U_{async} > (1 - U^{min}_{iso}). \tag{9}$$

Also from Case 3, we have

$$u^{min}_{m+1} \leq (1 - U^{min}_{iso}). \tag{10}$$

From Eq. (5), we have

$$U_{async} \leq (1 - U^{min}_{iso}). \tag{11}$$

So, when Eqs. (9), (10) and (11) are satisfied there may be performance loss.

Let us now derive the probability of performance loss. Assume that the values of minimum utilization of a new isochronous request, u^{min}_{m+1} and the total utilization of asynchronous requests, U_{async} vary uniform randomly between [0, p] and [0, q] respectively, where $0 < p < 1$, $0 < q < 1$ and $p \leq q$. The shaded area in Figure 1 shows the *region* in which the three conditions given in Eqs. (9), (10) and (11) are satisfied for a given $(1 - U^{min}_{iso}) = S$, $S \leq p \leq q$. The probability of performance loss is the ratio of area of the shaded isosceles right angle triangle to the area of rectangle of sides p and q. Thus, the probability of performance loss is $\frac{\frac{1}{2} \cdot S \cdot S}{pq} = \frac{S^2}{2pq}$. Due to space limitation we are not able to provide the analysis for other ranges of S, but can easily be done with suitable changes to the geometries in Figure 1. As S increases beyond q, this probability goes down and eventually goes to zero, when there is no intersection of the corresponding isosceles right angle triangle of sides S with the rectangle of sides p and q. Thus, if q is small, then performance loss occurs only at low values of S, i.e., at high values of U^{min}_{iso}.

TABLE I: Simulation Parameters

Parameter	Value
Arrival of requests	Poisson distributed with mean request arrival rate (λ) varied from 5 to 50 requests/BI
Maximum allocation duration (C^{max})	uniformly distributed between [10, 100] μs per BI
Allocation interval ratio (C^{min}/C^{max})	uniformly distributed between [0.5, 1]
Lifetime of isochronous requests	normally distributed with mean = 100 BI and stdev = 10 BI
Integer that defines integer fraction or integer multiple of BI as period of isochronous requests	uniformly distributed between [1, 5]
Duration of each run	1000 BI
BI duration	102 400 μs

TABLE II: Configuration Description

Configuration	Denoted as	Description
Config 0	async_0	No asynchronous requests, i.e., all arriving requests are isochronous
Config 1	async_30_1_9	30 % of arriving requests are asynchronous whose deadlines are uniformly distributed between [1, 9] BIs
Config 2	async_30_1_23	30 % of arriving requests are asynchronous whose deadlines are uniformly distributed between [1, 23] BIs
Config 3	async_50_1_9	50 % of arriving requests are asynchronous whose deadlines are uniformly distributed between [1, 9] BIs
Config 4	async_50_1_23	50 % of arriving requests are asynchronous whose deadlines are uniformly distributed between [1, 23] BIs

III. Simulation Results

A. Performance Metrics

The overall goal of our performance evaluation is to observe the impact of presence of asynchronous requests on the performance of isochronous requests. So, the performance metrics used to evaluate our ACIAR algorithm are defined with respect to isochronous requests.

- Acceptance Ratio (AR) : It is the ratio of the number of isochronous requests admitted to the total number of arriving requests.
- BI Utilization (BU): This is the fraction of a BI duration allocated to isochronous requests.
- Allocation Efficiency (AE): For an isochronous request T_i, it is defined as $\frac{T_i.C_i^{op} - T_i.C_i^{min}}{T_i.C_i^{max} - T_i.C_i^{min}}$. A higher AE translates to higher throughput for an application.
- Normalized Delay: The normalized delay of a job of an isochronous request is the difference of time instance of the end of allocation of the last fragment and the release time of the job, normalized with respect to the period of the request. It is an indicator of packet level delay.

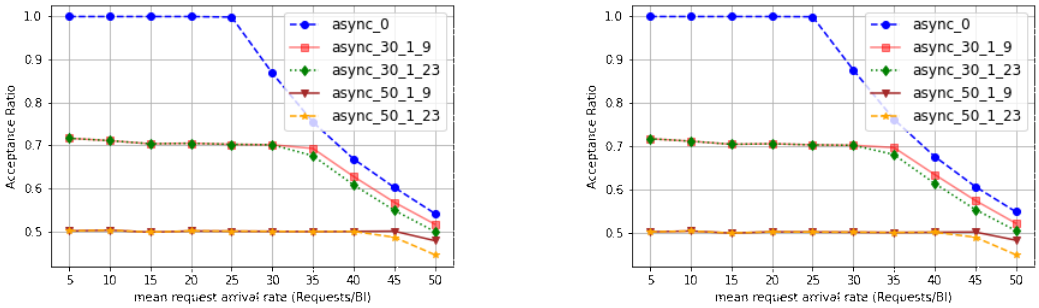
B. Simulation Experiment Design

We have developed a python based simulator that implements our ACIAR algorithm. Table I lists the simulation parameters used in our experiments and their values. Note that C^{min} is computed by multiplying the randomly generated allocation interval ratio (C^{min}/C^{max}) with randomly generated C^{max}. The experiments are carried out in two *scenarios* with respect to isochronous requests: i) when periods of all the isochronous requests are integer multiple of a BI (Scenario 1), ii) when periods of all the isochronous requests are integer fraction of a BI (Scenario 2). Scenario 1 simulates applications requiring low data rate (e.g., IoT applications), whereas Scenario 2 represents applications requiring high data rate and low delay (e.g., streaming video). In each scenario, we mix asynchronous requests with isochronous requests in five different ways giving rise to five configurations as listed in Table II. Config 0 acts as the baseline configuration against which the impact of asynchronous requests can be observed. Note that the mean deadline of asynchronous requests in Config 1 is lower than that in Config 2. Same is the case between Config 3 and Config 4.

C. Experiment Results

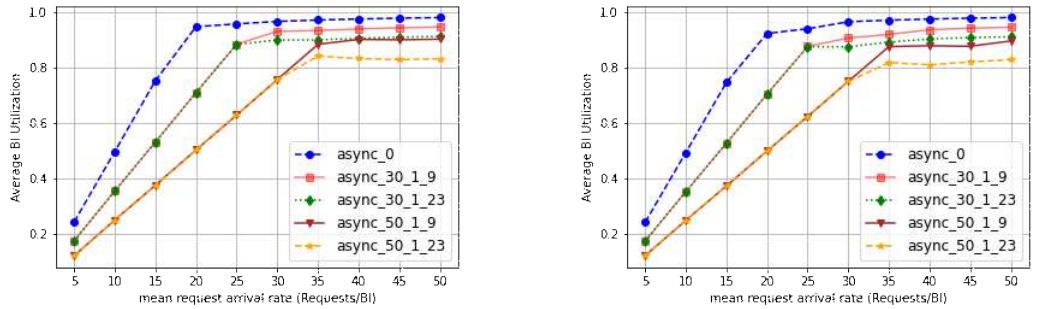
From Figure 2, in Scenario 1, at low λ, Config 0 achieves AR of 1.0. At low λ, system utilization is low and hence, the ACIAR is able to accept all requests. As λ increases towards a large value AR starts to drop due to high load. But in other configurations (Config 1 to Config 4), at low λ, AR of isochronous requests stay flat at a value below 1.0 until λ becomes large, when AR starts to drop. In these configurations isochronous requests share resources with asynchronous requests and hence, cannot attain AR of 1.0 at low λ. In fact, the value of AR at these low λ is approximately equal to the percentage of isochronous requests arriving to the system. For example, in Config 1, isochronous requests constitute 70 % of the requests and the AR value is approximately 0.7 at low λ values. This means that almost all isochronous and asynchronous requests are accepted into the system at low λ values. Comparing Config 1 with Config 2, we observe that there is not much difference of AR value between them at low λ. Mean deadline of asynchronous requests in Config 2 is higher than in Config 1. So, the asynchronous requests stay longer in the system in Config 2, but that does not affect the AR performance at low λ values since there are enough resources available. But at high λ values, Config 2 has slightly lower AR than Config 1 since overallocation of resources to asynchronous requests remains in effect for longer duration due to longer deadline of the asynchronous requests. AR of Config 3 and Config 4 is lower than Config 1 and Config 2 since they have higher percentage of asynchronous requests in the system. Relative AR performance across configurations in Scenario 2 is very similar to Scenario 1.

Observing Figure 3, BI utilization increases as λ increases in both scenarios and in all configurations. But after a certain λ value, BI utilization tapers off and stays flat. At those high λ values, the system is fully loaded (with the rest of the utilization taken up by asynchronous requests) and hence, there is no room to increase the BI utilization. BI utilization of Config 0 is the highest, since all the arriving requests are isochronous requests. In this configuration, isochronous requests are able to use up almost all the BI duration and



(a) Scenario 1 (b) Scenario 2

Fig. 2: Acceptance Ratio of Isochronous Requests vs. Mean Request Arrival Rate



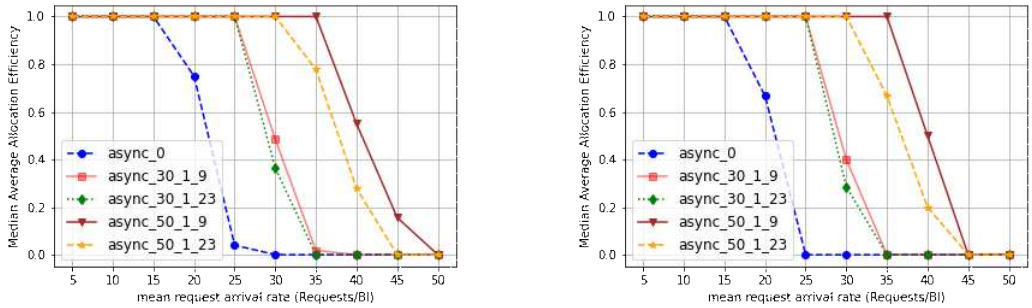
(a) Scenario 1 (b) Scenario 2

Fig. 3: Average BI Utilization of Isochronous Request vs. Mean Request Arrival Rate

achieve BI utilization close to 1.0 at high λ values. When the fraction of asynchronous requests is higher, BI utilization of isochronous requests is lower (e.g., BI utilization of Config 3 is lower than Config 1), since asynchronous requests occupy more BI duration. When the percentage of asynchronous requests is same between two configurations, but the mean deadline is longer, then BI utilization of isochronous requests is lower (e.g., BI utilization of Config 2 is lower than Config 1). When mean deadline of asynchronous requests is longer, they stay in the system for a longer duration and hence, isochronous requests occupy less BI duration. Relative performances of different configurations in Scenario 2 are very similar to Scenario 1.

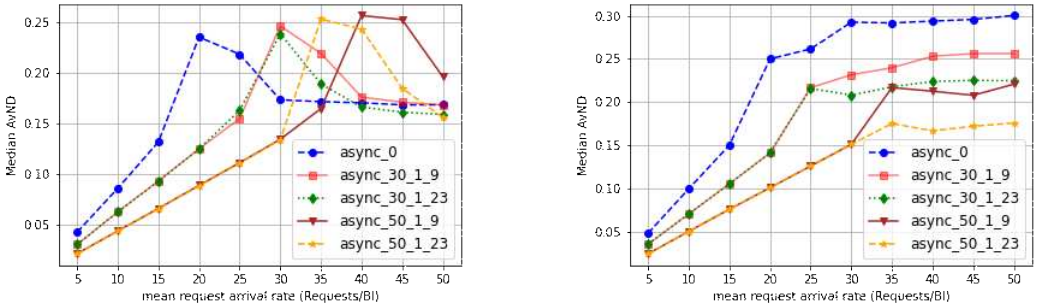
Figure 4 shows the change in median of average AE of the isochronous requests as mean arrival rate of requests (λ) increases. To compute the average AE of a given request, AE is computed in every period and averaged over its lifetime. Then the median of average AEs of the requests is plotted. For both scenarios, median AE starts at 1.0 but at high λ, it starts to drop and eventually becomes 0.0. At low λ, the system is lightly loaded and hence, isochronous requests are able to run at their C^{max} (AE = 1.0). But as load increases, the C^{op} starts to drop, eventually becoming C^{min}. Config 3 has higher AE than Config 1, since it has lower percentage of isochronous requests (and higher percentage of asynchronous requests). Asynchronous requests take less resources than isochronous requests (since they leave the system relatively quickly). Hence, each isochronous request gets more C^{op} in Config 3 compared to Config 1. In addition, in Config 3 there are less number of isochronous requests in the system than Config 1 and hence, more C^{op} could be allocated to each isochronous request in Config 3. Similarly, Config 4 has better AE than Config 2. When percentage of asynchronous requests is the same, but mean deadline is longer, then AE is lower, e.g., Config 2 has lower AE than Config 1 at high λ. In this case, since asynchronous requests stay longer in the system, isochronous requests get less resources. Config 0 has the worst AE since all the requests are isochronous which require more resources. Relative performance across different configurations is similar in Scenario 2.

Figures 5(a) and 5(b) show median of average normalized delay (AvND) of isochronous requests in the two scenarios as λ increases. For a given request, normalized delay is computed in every period and then averaged over its lifetime to obtain AvND. Then the median AvND of the requests is plotted. In both scenarios as λ increases initially AvND increases which is expected due to the increase in load. However, in Scenario 1, in every configuration, after AvND peaks, it starts to drop. Around when AvND peaks, AE starts to drop towards zero (see Figures 4(a)). Hence, C^{op} drops and moves towards C^{min}, which causes delay to decrease. Remember that the C^{min} and C^{max} are randomly generated as a "per BI" value. Then, for Scenario 1, they are scaled up to the multiple BI period, whereas for Scenario 2, they are scaled down to fraction BI period. Thus, when C^{op} decreases towards C^{min}, the change



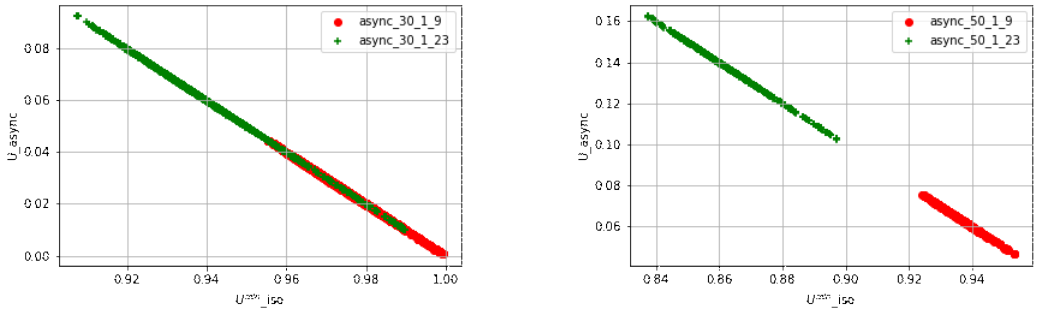
(a) Scenario 1 (b) Scenario 2

Fig. 4: Median of Average Allocation Efficiency of Isochronous Requests vs. Mean Request Arrival Rate



(a) Scenario 1 (b) Scenario 2

Fig. 5: Median AvND of Isochronous Requests vs. Mean Request Arrival Rate



(a) Scenario 1 (30 % asynchronous traffic) (b) Scenario 1 (50 % asynchronous traffic)

Fig. 6: Values of Utilization of Isochronous and Asynchronous Requests when Potential Performance Loss may Occur

is much higher in Scenario 1 than in Scenario 2. The large drop in C^{op} leads to lower delay. Hence, in Scenario 1, delay drops after peaking and later becomes almost flat when C^{op} equals C^{min}. In Scenario 2, the drop in C^{op} is not much when C^{op} moves towards C^{min}. Hence, the delay in Scenario 2, after peaking either drops slightly or continues almost flat in all the configurations. Comparing AvND across configurations in Scenario 1, we notice that Config 0 has the worst performance since the number of isochronous requests in the system is the highest in this configuration. Config 1 and 2 have the next highest number of isochronous requests and hence they have better performance than Config 0, but are worse than Config 3 and Config 4. Relative performances across configurations is very similar in Scenario 2. So, in general, we observe that at low load, delay performance of admitted isochronous request becomes better as more asynchronous requests are admitted to the system.

Figures 6(a) and 6(b) show scatter plot of values of (U_{iso}^{min}, U_{async}) when an isochronous request is rejected and all the three conditions given in Eqs. (9), 10) and (11) are satisfied in Scenario 1. This is the case when potential performance loss (of rejecting the isochronous request) may occur. In Figure 6(a) and 6(b) we see that those conditions are satisfied when U_{iso}^{min} is very high. When asynchronous percentage is 30 %, the situation starts to arise at a slightly higher U_{iso} than when asynchronous percentage is 50 %. This is because, with lower asynchronous request load, U_{iso}^{min} has to be higher so that Eq. (9) is satisfied. When mean deadline of asynchronous

requests increases (while the percentage remains the same), the performance loss starts at a lower U_{iso}^{min}. With longer deadlines, the asynchronous requests stay longer and effectively increase the asynchronous utilization. Hence, the performance loss condition of Eq. (11) is satisfied at a lower U_{iso}^{min}. The performance in Scenario 2 is very similar, but we do not present the graphs due to space limitation.

IV. Related Work

IEEE 802.11ad channel access has been analytically modeled by researchers. SP and CBAP mode of channel access has been modeled as a 3D Markov chain in [8]. A Markov chain based analytical model for CBAP allocation is presented in [9]. The model takes the presence of SP allocation, deafness and hidden node problems due to directional antennas into account during CBAP. A study of worst case delay of packets transmitted through SP allocation using an analytical model for SP access is presented in [10]. They also formulate a method for optimal allocation sharing between SP and CBAP. Scheduling of a multimedia flow that uses SP channel access is analytically modeled in [11]. The model accounts for channel errors.

Very little work has been reported in the field of experimental study of admission control and scheduling of SP and CBAP channel allocation. A max-min fair algorithm for admission control and scheduling of isochronous traffic using SP allocation is presented in [4]. But it does not handle asynchronous traffic. Two application scenarios considered in that work are very simple. In the first scenario, all applications have the same parameter values whereas in the second scenario, the applications choose one set of parameter values out of only two predetermined set of values. It considers only the case when the periods of isochronous requests are integer fraction of a BI. Their algorithm has a cubic run time complexity in terms of number of requests. In contrast, our algorithm has a linear run time complexity and can handle both isochronous and asynchronous requests and do not have any restrictions in terms of their periods. A reinforcement learning (RL) based scheduling of SP allocation is studied in [12]. Using Q-learning paradigm and through interaction with the network deployment scenario, it finds the optimal SP duration. MAC layer queue length in terms of number of packets is used as *states*. A *reward* is computed as a function of number of correctly received packets and an *action* is taken. Three admission control algorithms for isochronous traffic that are fair and compliant with the IEEE 802.11ad standard are presented in [2]. It also proposes an EDF based scheduler which guarantees appropriate SP durations to the admitted isochronous requests before their respective deadlines. Unlike the algorithms proposed in [4], the algorithms in [2] run with a linear run time complexity even when the requests choose any arbitrary values for their parameters. An admission control and scheduling algorithm for isochronous traffic in IEEE 802.11ad MAC that accounts for guard time overhead was reported in [13].

V. Conclusion

We presented an admission control algorithm, ACIAR, which admits both isochronous and asynchronous requests in an IEEE 802.11ad MAC with a linear run time complexity and is proportionally fair. It uses EDF scheduling algorithm to allocate SP duration to the requests in a BI. We presented performance results of the ACIAR algorithm using different performance metrics and used different percentage mix of asynchronous requests in the system to observe the impact of asynchronous requests on the performance of isochronous requests. It was observed that the number of isochronous requests accepted as well as the BI utilization of isochronous requests go down when they share the system with more asynchronous requests. However, the performance of those smaller number of accepted isochronous requests improves with respect to other metrics such as allocation efficiency (which translates to throughput) and delay due to this sharing. We derived the conditions when performance loss, because of considering asynchronous request as periodic in ACIAR algorithm, may occur.

References

[1] "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," 802.11 Working Group of the LAN/MAN Standards Committee of the IEEE Computer Society, Dec. 2016.

[2] A. Sahoo, W. Gao, T. Ropitault, and N. Golmie, "Admission control and scheduling of isochronous traffic in ieee 802.11 ad mac," in *Proceedings of the 24th International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2021, pp. 125–134.

[3] C. L. Liu and J. W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment," *Journal of the Association for Computing Machinery*, vol. 20, no. 1, pp. 46–61, January 1973.

[4] M. Lecci, M. Drago, A. Zanella, and M. Zorzi, "Exploiting scheduled access features of mmwave wlans for periodic traffic sources," in *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*. IEEE, 2021, pp. 1–8.

[5] K. Jeffay, D. F. Stanat and C. U. Martel, "On Non-Preemptive Scheduling of Periodic and Sporadic Tasks," in *IEEE Real-Time Systems Symposium (RTSS)*, December 1991, pp. 129–139.

[6] S. R. Thuel and J. P. Lehoczky, "Algorithms for Scheduling Hard Aperiodic Tasks in Fixed-Priority Systems Using Slack Stealing," in *RTSS*, 1994, pp. 22–33.

[7] H. Chetto and M. Chetto, "Some Results of the Earliest Deadline Scheduling Algorithm," *IEEE Transactions on software engineering*, vol. 15, no. 10, pp. 1261–1269, 1989.

[8] Q. Chen, J. Tang, D. T. C. Wong, X. Peng, and Y. Zhang, "Directional Cooperative MAC Protocol Design and Performance Analysis for IEEE 802.11ad WLANs," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 6, pp. 2667–2677, 2013.

[9] C. Pielli, T. Ropitault, N. Golmie, and M. Zorzi, "An Analytical Model for CBAP Allocations in IEEE 802.11ad," *IEEE Transactions on Communications*, 2020.

[10] C. Hemanth and T. Venkatesh, "Performance Analysis of Service Periods (SP) of the IEEE 802.11ad Hybrid MAC Protocol," *IEEE Transactions on Mobile Computing*, vol. 15, no. 5, pp. 1224–1236, 2015.

[11] E. Khorov, A. Ivanov, A. Lyakhov, and V. Zankin, "Mathematical Model for Scheduling in IEEE 802.11ad Networks," in *2016 9th IFIP Wireless and Mobile Networking Conference (WMNC)*. IEEE, 2016, pp. 153–160.

[12] T. Azzino, T. Ropitault, and M. Zorzi, "Scheduling the Data Transmission Interval in IEEE 802.11ad: A Reinforcement Learning Approach," in *2020 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2020, pp. 602–607.

[13] A. Sahoo, W. Gao, T. Ropitault and N. Golmie, "Admission control and scheduling of isochronous traffic with guard time in IEEE 802.11ad MAC," *IEEE Transactions on Mobile Computing*, pp. 1–10, 2022, doi:10.1109/TMC.2022.3207969.