Demand-Aware Centralized Traffic Scheduling in Wireless LANs

Sangyup Han
KAIST

Myungjin Lee
University of Edinburgh

Myungchul Kim
KAIST

***Abstract*—A heavy deployment of IEEE 802.11 Wireless LANs and limited number of orthogonal channels make lots of Access Points (APs) overlap their interference regions, which greatly increases interferences between APs and stations. In order to cope with the performance degradation caused by the interferences, we propose CO-FI, a centralized Wi-Fi architecture that effectively coordinates downlink transmissions by APs and improves network performance in terms of throughput and end-to-end delay. CO-FI adaptively allocates time slots for APs and stations based on both traffic demands on the stations and a conflict graph that represents interference relationships among the devices. The scheme allows APs in exposed node relationship to use the channel simultaneously by setting the same backoff time. It also effectively avoids downlink conflicts created by hidden node and non-hidden/non-exposed node, by allocating non-overlapping time slots to interfering stations. To implement these adaptive traffic schedules, we design CoMAC, a hybrid MAC protocol at APs. Our evaluation results show that when APs are densely deployed and the network is highly loaded, the scheme achieves 3-5 times more throughput gain than Centaur, a state-of-the-art scheme while its end-to-end delays are 10-90% lower than those of Centaur and CSMA/CA.**

I. INTRODUCTION

IEEE 802.11 Wireless LAN (WLAN) is one of the most popular wireless communication technologies developed so far. Its tremendous success has led to the dense deployment of WLANs almost everywhere. However, the high density also incurs interferences more frequently among wireless Access Points (APs) and devices (or stations) [1]. Hence, more APs may do more harm than good, and hamper the optimal performance of WLANs [2].

In fact, the interference problem in WLANs is one of well-studied topics in the literature. A large body of research work [3], [4], [5] has focused on reducing interference level to improve stations' throughput. In [5], it is seen as a channel allocation problem, and several graph coloring algorithms are explored. In [3], dynamic transmission range control is attempted. However, the heavy deployment of WLANs still creates various interference situations and makes those approaches less effective. Consequently, recent approaches [6], [7], [8] explore centralized traffic scheduling in order to fundamentally minimize the degree of interference.

However, centralized scheduling in general entails high scheduling complexity. Existing solutions therefore trade scheduling granularity for reduced complexity. For instance, Centaur [6], one of the state-of-the-art approaches, performs centralized scheduling only for traffic of hidden and exposed nodes whereas it delegates the scheduling of traffic for non-hidden/non-exposed nodes to Distributed Coordination Function (DCF) of CSMA/CA. Thus, contentions can hurt throughput for traffic destined to the non-hidden/non-exposed nodes. Worse, in the presence of automatic rate adaptation [9], [10], contentions may force selection of lower rates more often, which may further exacerbate performance.

Such trade-off of the state-of-the-art solution eliminates the possibility of improved throughput through precise scheduling. As such, we take into account all of the interference types including non-hidden/non-exposed node for traffic scheduling. To amortize the increased complexity, we only focus on batch-scheduling of high-volume traffic as scheduling low-volume traffic well does little for overall performance improvement.

In this paper, we present *Coordinated* Wi-Fi (CO-FI) that achieves high throughput and low scheduling complexity in WLANs administered by a single authority. CO-FI is designed in a way that a centralized controller computes frame transmission schedules for each AP, and APs run a hybrid MAC protocol called *CoMAC* that can select DCF and Time Division Multiple Access (TDMA) modes flexibly. CoMAC runs in TDMA mode to transmit traffic in a batch fashion scheduled by the controller whereas it runs in DCF mode for transmitting low-volume traffic. Our scheme only schedules downlink traffic as the volume of downlink traffic takes a dominant portion in WLANs [11], [12], [13]. That is, stations access wireless medium in a typical CSMA/CA manner for uplink transmission.

In summary, this paper makes the following contributions:

- We present CO-FI, a novel centralized traffic scheduling mechanism for WLANs that effectively coordinates downlink transmission of frames to stations. The scheme employs demand-aware traffic scheduling. In the scheme, traffic to bandwidth-hungry stations is precisely scheduled while transmission of low-volume traffic takes place opportunistically. This allows CO-FI to keep scheduling overhead low as a small number of stations need to be scheduled on average.
- We design CoMAC, a *hybrid MAC scheme* that can elastically switch between DCF and TDMA. CoMAC works in TDMA mode when the transmission of high-volume traffic strictly follows the schedule of the centralized scheduler. In contrast, DCF mode is activated for opportunistically scheduling the transmission of low-

ISBN 978-3-901882-83-8 © 2016 IFIP

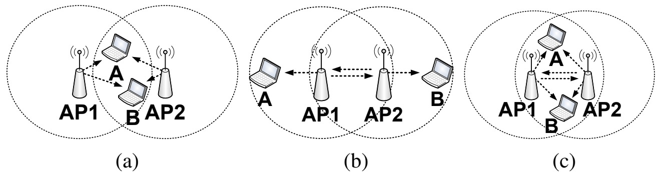


Fig. 1: Types of interferences: (a) Hidden-Node (HN), (b) Exposed-Node (EN) and (c) Non-Hidden/Non-Exposed Node (NHNEN). *Stations A* and *B* are associated with *AP1* and *AP2*, respectively.

volume traffic, which prevents starvation. Because the scheme supports both modes, potential schedule conflicts in TDMA mode can be addressed using DCF. This not only allows our scheme to avoid complex rescheduling but also makes it robust to errors in time synchronization and traffic demand estimation.

- Our extensive simulation results demonstrate that Co-Fi outperforms Centaur, the most well-known solution, and CSMA/CA when APs are densely deployed and the network is highly loaded. Specifically, Co-Fi achieves 3-5× higher throughput than both schemes and its end-to-end delays are 10-90% lower than those of Centaur and CSMA/CA.

The remainder of this paper is organized as follows. Section II introduces the basic concepts necessary when discussing the design of Co-Fi in Section III. In Section IV, we present evaluation results. Section V discusses related work before we conclude in Section VI.

II. Preliminaries

In this section, we discuss three primary concepts—*interference types*, *time window* and *time slot*—that are the basis in devising our scheme.

Interference types. We first explore different characteristics of interferences between wireless links. For this, we adopt a well-known data structure called *Conflict Graph* [14]. A vertex in a conflict graph represents an AP or a station, and a directed edge between two vertices means a wireless link. If there is a wireless link (same as an edge in a conflict graph), it means that a signal from an AP (or station) can successfully be transmitted to the other AP (or station).

Figure 1 illustrates three types of interferences. *Stations A* and *B* in the figure are associated with *AP1* and *AP2*, respectively. In addition, the wireless links outgoing from the APs are only shown in Figure 1 since we only consider the interferences caused by downlink transmissions. A wireless link from node i to node j is denoted as L_{ij}. For instance, L_{1A} represents the wireless link from *AP1* to *Station A*; other wireless links are denoted in the same manner.

Although the concepts of hidden-node and exposed-node problems are well known, the way to identify them from a conflict graph varies across studies [14], [15]. Thus, we slightly modify them and use the following equations when the edge set, *E*, of a conflict graph is given:

- Hidden-Node (HN) interference:

$$\begin{aligned}&\{L_{12},\, L_{21}\} \not\subset E\, \textbf{\textit{and}}\, \{L_{1A},\, L_{2B}\} \subset E \\ &\qquad \textbf{\textit{and}}\, (L_{1B} \in E\, \textbf{\textit{or}}\, L_{2A} \in E),\end{aligned} \tag{1}$$

- Exposed-Node (EN) interference:

$$\begin{aligned}&(L_{12} \in E\, \textbf{\textit{or}}\, L_{21} \in E)\, \textbf{\textit{and}}\, \{L_{1A},\, L_{2B}\} \subset E \\ &\qquad \textbf{\textit{and}}\, (L_{1B} \notin E\, \textbf{\textit{and}}\, L_{2A} \notin E),\end{aligned} \tag{2}$$

- Non-Hidden/Non-Exposed Node (NHNEN) interference:

$$\begin{aligned}&(L_{12} \in E\, \textbf{\textit{or}}\, L_{21} \in E)\, \textbf{\textit{and}}\, \{L_{1A},\, L_{2B}\} \subset E \\ &\qquad \textbf{\textit{and}}\, (L_{1B} \in E\, \textbf{\textit{or}}\, L_{2A} \in E).\end{aligned} \tag{3}$$

Note L_{uv} represents an edge from vertex u to vertex v. We use number for u and v to denote APs and use alphabets to denote stations. In addition, further notice that Eqs. 1, 2, and 3 cover the case that only one of the two APs senses the other AP, which is caused by the asymmetric nature of wireless medium. If edges meet none of the above conditions, we treat them *as if there is no interference among them*. These definitions are applied to more complex WLANs through pair-wise comparisons of edges iteratively.

Issues with these interferences: The HN and NHNEN interferences result in collision at APs when simultaneous transmissions take place from APs to stations. This therefore causes retransmissions and even frame drops. While DCF may mitigate the impact of these interferences, not all collisions can be avoided. Moreover, frame collisions may make the APs decrease their PHY transmission rate, which results in performance degradation. On the other hand, if stations experience EN interference, their associated APs can benefit from simultaneous transmissions and achieve improved throughput. However, if an AP senses the signal of other APs, it defers its transmission and fails to exploit the EN interference.

Time window and slot. A time window is a basic unit of scheduling frame transmissions, and a time slot is a constituent of a window. Note that a frame transmission can span multiple consecutive time slots due to a low transmission rate. In our paper, the duration of a window, Δ, is set to 20 ms, and each window consists of 800 slots. Thus, one slot corresponds to 25 μs. Whilst both variables are flexibly configurable, we choose them empirically while running simulations.

III. Co-Fi Design

We now design Co-Fi, a centrally-coordinated WLAN architecture. In Co-Fi, a centralized controller coordinates traffic transmission schedules of APs in order to maximize throughput and to minimize end-to-end delay in the WLANs. We first present an overview of Co-Fi, and then describe each component that constitutes the architecture.

A. Overview

Co-Fi adopts a centralized coordination model where a controller precisely schedules traffic transmission timings of APs. Under this model, APs communicate with the controller to form a control loop for traffic transmission coordination.

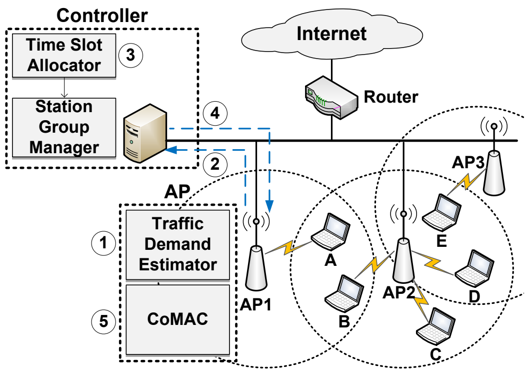


Fig. 2: Overview of Co-Fi. The overall procedure is to: (1) estimate future traffic demands to stations; (2) inform the controller of the traffic estimates for stations; (3) compute time slot allocation schedule for the traffic among interfering stations; (4) distribute the schedule to APs from the controller; (5) transmit frames either based on time slots allocated or opportunistically if no time slot is allocated.

Specifically, the controller dictates when each AP can transmit, and the APs abide by the controller's instruction. Co-Fi adaptively allocates time slots for APs while being aware of the amount of workloads that arrive at each AP. In addition, Co-Fi addresses potential schedule conflicts by leveraging existing CSMA/CA. We call this strategy CoMAC which is discussed in Section III-D.

We describe Co-Fi's working mechanism through a simple scenario illustrated in Figure 2. Suppose that the amount of traffic to *Stations A* and *B* is 120 Kbits and 60 Kbits, respectively, over a window Δ. *AP1* and *AP2* estimate future traffic demands by combining the size of existing packets in their buffer with the amount of incoming traffic (**Step 1** in Figure 2). The APs next ship these estimates to the controller (**Step 2**). The controller then allocates time slots in proportion to these estimates as the two APs compete for the same wireless medium (**Step 3**). In Figure 2, because the two links are in an HN relationship, the controller allocates non-overlapped time slots in proportion to each AP's demand (*i.e.*, 533 slots to *A* and 267 slots to *B*). This prevents the APs from simultaneous transmission. In addition, if the estimated traffic volume to a station is lower than a threshold, ψ (further discussed in Section III-D), the station is excluded from scheduling and allowed to do the typical opportunistic medium access via CSMA/CA. This effectively reduces scheduling complexity. The controller distributes the allocation information to APs (**Step 4**). The APs then take one of the following two actions (**Step 5**): i) if a destination station is allocated to some time slots, a frame to the station can only be transmitted within its time slots; ii) for those stations without allocated time slots frames are transmitted opportunistically via CSMA/CA.

One key advantage of our scheme is that it requires no modification of the stations, which renders it practical in deploying it into the existing and future wireless networks. In the rest of this section, we discuss how we design each component of Co-Fi in Figure 2.

B. Traffic Demand Estimator at APs

APs estimate how much amount of traffic the APs should transmit for a given station at the next time window. At the end of every time window, the APs conduct the process for each station associated with them. They then send the estimates to the controller for a centralized time slot allocation process.

The APs maintain a table that consists of the following column elements: (W_i, *dstMAC*, T_i) where W_i denotes time window i, *dstMAC* means the MAC address of a destination station, and T_i is a total of traffic demand (in bytes) to the destination in W_i. Whenever the APs see a new frame, APs extract *dstMAC* and frame size f and update T_i with f. (*i.e.*, $T_i \leftarrow T_i + f$). For a given *dstMAC*, we refer to the information stored in the table and estimate the future demand as follows:

$$FD_i^{dstMAC} = \min(MA_i + D_i, Tx_i \times \Delta), \qquad (4)$$

where MA_i is an exponential moving average of incoming traffic to the station at W_i, D_i is the traffic amount (in bytes) in the buffer for the station at W_i, and Tx_i is the current PHY transmission bitrate for the station at W_i. Note that in (4), Δ is converted from millisecond to second. These three variables (MA_i, D_i, Tx_i) are maintained on a per-station basis.

In the min function of the equation, the left-hand side term indicates the amount of traffic that the AP should transmit at the next time window. However, when Tx_i is low, the AP cannot transmit all the traffic within the next time window. Therefore, we put the right-hand side term as an upper bound. MA_i is calculated using the following equation:

$$MA_i = \begin{cases} \alpha \cdot T_i + (1-\alpha) \cdot MA_{i-1}, & \text{if } T_i \neq 0 \\ (1-\alpha) \cdot MA_{i-1}, & \text{otherwise} \end{cases}, \qquad (5)$$

where T_i is the amount of traffic received during time window i, and α is the coefficient that represents the degree of weighting the current traffic. We set α to 0.8 in this paper.

While in principle APs can report the computed demands at every window ($\Delta = 20\,ms$), in practice we have APs report $\max(FD_i^{dstMAC}, FD_{i-1}^{dstMAC})$ every other window (so, 40 ms). We do this because we found that reporting the traffic demand at every window sometimes became unstable, and reporting at every other window achieved the highest throughput (always 3% greater than the former in our test scenarios).

C. Controller functions

The Co-Fi controller has two core functions: time slot allocation and station grouping. In addition, the controller synchronizes time among APs by using Network Time Protocol (NTP). This protocol is known to make a few millisecond synchronization precision possible in the wired local networks [16]. To tolerate that level of synchronization error, in our design we use a large (i.e., $20ms$) window.

Time Slot Allocator. Figure 3 illustrates the time slot allocation procedure. At first, APs estimate traffic demands for stations and send them to a controller every $2 \cdot \Delta$ (**Step 1** in the figure). The controller receives the estimates, as the form of (station's MAC address, traffic estimate), from the APs for

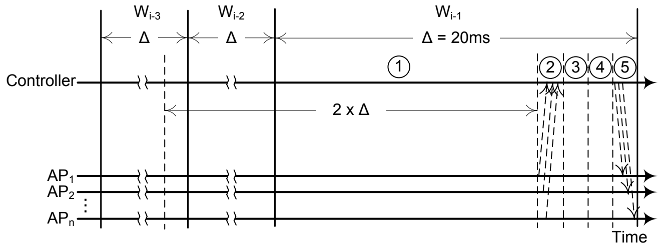


Fig. 3: Slot allocation scheduling procedure: (1) APs estimate traffic demand for stations for $2{\cdot}\Delta$ period, (2) the controller receives traffic demand estimates from the APs for $\tau\ ms$, (3) the controller retrieves station groups via station group manager module, (4) it allocates time slots for stations on a per-group basis, and (5) it sends slot allocations to the APs.

a fixed period of τ (**Step 2**). We empirically set $\tau = 2\,ms$ to keep high responsiveness and network performance. The demand estimation that arrives later than $2ms$ is ignored. This is a reasonable value in local area networks where the end-to-end delay between the controller and APs can be on the order of a few hundreds of microseconds. In our campus network, we observe round-trip times are almost always less than $1\,ms$.

The allocator then hands over the MAC addresses of the stations to station group manager. Next, the manager clusters stations into groups based on the interference types presented in a conflict graph (**Step 3**). For instance, when a wireless link from an AP to a station is in a relationship of interference with another link, two stations belong to the same station group. Transmissions to the stations should be scheduled with non-overlapped time slots. If stations do not belong to a group, they do not interfere with any other stations; such stations acquire full access to medium within that window.

The allocator sorts the station groups in a decreasing order of their total demands (*i.e.*, the sum of the traffic demands of all stations in a group) and allocates time slots to stations on a per-group basis (**Step 4**). Hence, time slots are first allocated to the group with maximum traffic demand. This allows the wireless network to maximize the total throughput. The detailed algorithm for allocating time slots to the sorted groups is given in Algorithm 1. Note that a station can be a member of multiple groups. If time slots for the station were already allocated, it is excluded from time slot allocation (at line 8 in Algorithm 1). As a final step (**Step 5**), the allocation information is disseminated to the APs which work in TDMA mode for the next two window times (*i.e.*, $40ms$ in our paper). During TDMA mode, APs stick to current allocations until a new allocation is fetched from the controller.

Station Group Manager. The controller determines the group to which a station should belong, by leveraging interference relationships among stations in a conflict graph in Section II. The manager constructs a conflict graph using an algorithm in [15] during a booting time of the controller. Note that our system does not rely on a particular conflict graph construction algorithm, and thus other techniques, such as [17], can also be used as an alternative.

Algorithm 1 Time slot allocation

```
 1: procedure ALLOCATOR( $\Omega$ )
 2:     ▷ $\Omega$: a set of sorted station groups
 3:     ▷ $W$: no. of total slots per window
 4:     if $\Omega$ is equal to $\phi$ then
 5:         return
 6:     $G \leftarrow$ PickNextGroup( $\Omega$ )                ▷ $G$: a group
 7:     for all $s \in G$ do                                ▷ $s$: a station
 8:         if IsAlreadyAllocated( $s$ ) then
 9:             Go to line 7
10:         $k \leftarrow s$.Demand / $G$.Demand $\times W$
11:         $n \leftarrow 0$
12:         $A \leftarrow$ APof( $s$ )    ▷ $A$: AP that $s$ is connected to
13:         while $n \leq k$ do
14:             $m \leftarrow$ GetFirstUnallocatedSlot( $G$ )
15:             Link( $A$ to $s$ ).add( $m$ )
16:             $n \leftarrow n + 1$
17:     ALLOCATOR( $\Omega - G$ )                    ▷ Call recursively
```

The procedure for grouping stations is simple. (a) Given edge e_i (*i.e.*, a downlink or a link from an AP to a station) from a conflict graph, the algorithm selects another edge e_j ($i \neq j$) from the graph and checks the interference relationship between the two, based on the definitions of interferences in Section II. (b) If the links create either the HN or NHNEN interference, they are grouped together. (c) The manager chooses a next edge and repeats this process until all other edges are tested against e_i. These three steps ((a)-(c)) are executed for all edges in the conflict graph. Due to the asymmetric nature of wireless medium, the interfering station set can be a subset of another one, and we discard such a set. The resulting station groups therefore are not a subset of any other sets. However, some stations can belong to more than one groups because they can have different interference relationships with other stations. As already discussed, the controller prevents such stations from getting time slots more than once (see Algorithm 1). The complexity of our station grouping procedure is $O(n^2)$ where n is the number of wireless downlinks in the conflict graph.

The overall overhead of the controller system is low as updating the conflict graph can be done incrementally during Step 1 in Figure 3 and the allocation algorithm at Step 4 is straightforward. For EN interference, we exploit the Centaur's mechanism [6]; *i.e.*, APs use the same backoff time for the EN stations to them while keeping CSMA/CA being enabled. The controller informs the APs of a list of those EN stations.

D. CoMAC at APs

We design CoMAC that can elastically support TDMA and CSMA/CA while keeping it backward compatible with CSMA/CA. Specifically, CoMAC at APs runs atop CSMA/CA and informs CSMA/CA when and to which station the APs can transmit frames. Therefore, except for selecting a destination station, the underlying CSMA/CA is unmodified. Note that an AP enables TDMA mode only if it has at least one associated station for which some time slots are allocated.

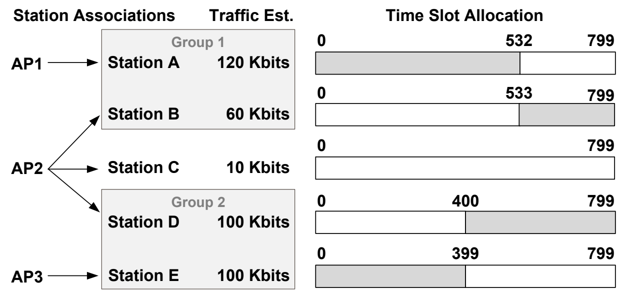


Fig. 4: An example of time slot allocation. A dark region represents allocated time slots.

Upon receiving a time slot allocation schedule from the controller, the APs begin to transmit frames as dictated in the schedule. However, there is a possibility that an AP may have a conflicting schedule for its stations or needs to share time slots with unscheduled stations that wish to receive a low volume of traffic. The AP handles such cases by selecting stations in a round-robin fashion.

To understand how the round robin strategy works, consider a scenario in Figure 4 where *Station A* is associated with *AP1*, *Stations B, C* and *D* with *AP2*, and *Station E* with *AP3*. The controller places *A* and *B* into Group 1 and *D* and *E* into Group 2 because each pair of the stations has an HN interference relationship (see Figure 2). Next, it computes a slot allocation schedule on a per-group basis (see Section III-C), which leads to an overlap of time slots for *B* and *D*. *C* has no allocated time slot because the estimated traffic volume for the station within Δ is less than $\psi = 13$ Kbits. Note that the value of ψ is empirically chosen and can be easily tuned depending on Δ. Given the scenario, since *B* and *D* own time slots within the window, and *C* does not, *AP2* can send frames to *C* in its round-robin turn between slot 0 and 799. On the other hand, *D* can occupy time slots 400-532 whereas *B* and *D* should take turns to share slots 533-799.

Once a station is selected, CoMAC calculates transmission duration for the first frame of that station in the buffer. If the transmission duration resides in the allocated time slots for the station, CoMAC dispatches the frame to CSMA/CA module for transmission. Otherwise, CoMAC reselects another station.

E. Implementation Issues

We base our evaluation on simulations, and leave a real implementation as future work. Hence, we here briefly discuss how to implement Co-Fi. We envision that implementing our controller functions on an OpenFlow controller [18] is feasible. Implementing CoMAC at APs can be done through modifying the source of wireless device drivers. The MadWiFi driver is a sensible starting point, which is a widely used driver in the literature [7], [19]. MAClets [20] can also be a viable platform as it supports modification of medium access control operation. Another implementation issue is in constructing a conflict graph. We can use the active probing method in [15] or the passive method in [17]. The controller can instantly update a conflict graph using the probing method; however, it can be more complex than the passive method.

IV. Evaluation

We now evaluate Co-Fi in this section. To demonstrate its benefits, we comprehensively perform simulations using QualNet [21] and present the results. The evaluation mainly consists of two parts: i) case 1: impact of each interference type on performance in simple WLANs; and ii) case 2: performance in a realistic large-scale WLAN setup. We begin our discussion with simulation setup.

A. Basic Simulation Setup

Approaches. To evaluate the efficacy of Co-Fi, we compare Co-Fi with CSMA/CA and Centaur.

Traffic workloads. We first use two application-level protocols (HTTP that uses TCP and Constant Bit Rate (CBR) that uses UDP). The HTTP traffic is generated using the generation model based on the trace-driven packet analysis [22]. These traffic sources are used for investigating the impact of each interference type. For more realistic simulations, we generate TCP sessions by modeling their arrivals as a Poisson process. The exact number of TCP sessions per second is controlled by an arrival rate. Every time a TCP flow is created, its size is determined probabilistically by exploiting empirical measurement data on flow size distribution (see the CDF curve on download size in Fig. 3(a) in [11]).

Interference types. We take into account all three kinds of interferences in our simulations: HN, EN and NHNEN. For simulations in Section IV-B, we generate these interferences one by one in an isolated fashion. In contrast, all three types coexist in a realistic setting in Section IV-C.

MAC protocols. As a wired MAC protocol, we employ a Gigabit Ethernet to connect all APs and controller, and use the propagation delay of 0.5 μs. As for a wireless MAC protocol, we use IEEE 802.11n and enable Auto Rate Fallback (ARF), a rate adaptation algorithm used by CSMA/CA [9]. Finally, all the APs and stations use the same channel in the 2.4 GHz band across all simulations.

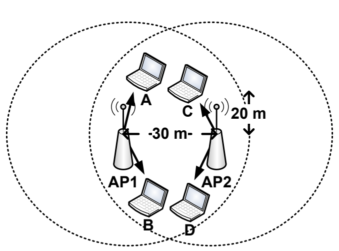
B. Influence of Each Interference Type

Configuration: We first identify what kinds of interferences our scheme can handle well. We create three controlled scenarios (NHNEN interference only, HN interference only, and EN interference only) as shown in Figure 5. For all the scenarios, the carrier-sense regions of all APs are illustrated in a reduced scale; but, all the interference relationships are preserved. The association relationships between APs and stations are presented as arrows in the figure.

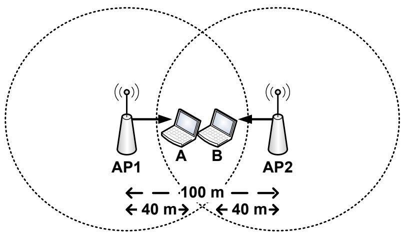
Each application across all the scenarios was simulated 50 times by varying the random seed value in the QualNet simulator. The random seed affects not only the characteristics of applications, such as the number of items per Web page, but also the characteristics of a wireless environment.

Results: We evaluate the performance of our scheme in terms of throughput and end-to-end delay.

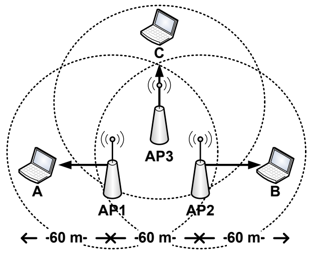
1) Throughput performance. Figure 6 shows the average throughput of HTTP traffic over 50 simulation runs. Under



(a) Non-Hidden/Non-Exposed Node (NHNEN)



(b) Hidden-Node (HN)



(c) Exposed-Node (EN)

Fig. 5: Basic simulation scenarios.

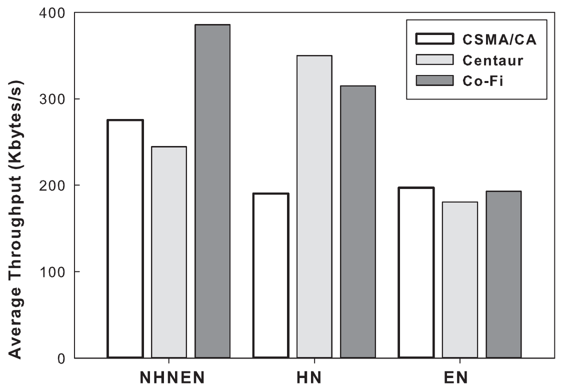


Fig. 6: Average throughput of HTTP (TCP) downlink traffic for each basic scenario.

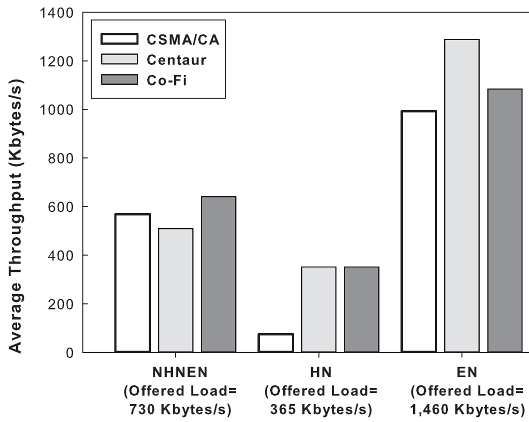


Fig. 7: Average throughput of CBR (UDP) downlink traffic for each basic scenario.

the NHNEN scenario, Co-Fi achieves the highest throughput because it makes sure that frame collisions at stations occur less than other schemes. Thus, less collisions prevent ARF from slowing down a PHY transmission rate. On the other hand, other schemes face more collisions, and ARF cannot help but reduce the Tx rate, resulting in poor performance. Furthermore, Centaur obtains lower throughput than CSMA/CA. Because the Centaur controller does not forward the frames that require more time than the remaining time in a time window, frames are not often scheduled at the fringe of the current window and the next, leading to a throughput loss.

In the HN scenario, CSMA/CA achieves only about 190 KB/s as CSMA/CA is susceptible to the HN problem. In contrast, Co-Fi and Centaur achieve at least 1.68× higher throughput than CSMA/CA. Between the two, Centaur slightly works better than Co-Fi. As Co-Fi does not schedule low volume traffic in a TDMA fashion, this may cause HN interferences to other stations. We trade this level of throughput loss for low scheduling complexity in Co-Fi.

Under the EN scenario, there is no much performance difference among all three schemes. The main cause of this result is that TCP generates two-way packet streams (one for data from AP to station and the other for ACK and HTTP GET from station towards AP), and uplink streams interfere with the other downlink streams. For example, the uplink stream by station *B* and the downlink stream by *AP1* collide with each other at *AP2* due to the omnidirectional characteristic of wireless signals. Therefore, the HTTP server sends data traffic to each station sequentially; hence, there is little chance for APs to simultaneously transmit data packets in order to leverage the EN relationship.

Figure 7 shows the average throughput of CBR traffic under each controlled interference scenario. For each scenario, we use different CBR rates. Across all scenarios, both Co-Fi and Centaur perform better than CSMA/CA. As opposed to the HTTP case, Co-Fi and Centaur provide a clear performance benefit under the EN scenario. As already discussed, TCP generates traffic in both directions, and ACK and HTTP GET traffic can be a major source of interference. On the other hand, UDP does not face such an issue. In most cases, the performance of Co-Fi is comparable to that of Centaur.

2) End-to-End delay performance. Figure 8 shows the average end-to-end delay of HTTP traffic for each scenario. Co-Fi achieves the lowest end-to-end delays under the NHNEN scenario and obtains an end-to-end delay similar to that Centaur under the HN case. On the other hand, there is no visible difference in end-to-end delay in the EN case.

The simulation results with CBR traffic is somewhat different from those of HTTP traffic (see Figure 9). Before further discussion, note that in the figure, the offered CBR rates are different across interference types. The reason we vary the CBR rate is because the impacts of different interference types on delay (and throughput too) are indistinguishable among different schemes without increasing the traffic rate in the sequence of HN, NHNEN and EN. Thus, absolute delays

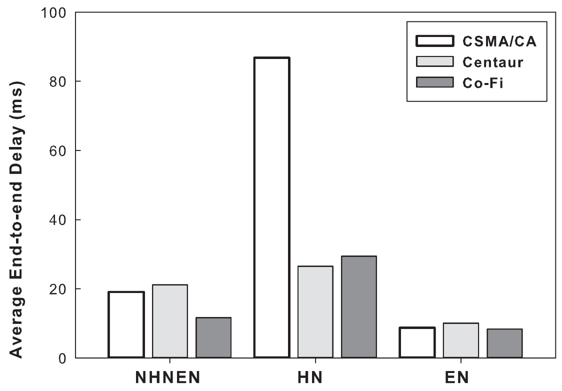


Fig. 8: Average end-to-end delay of HTTP (TCP) downlink traffic for each basic scenario.

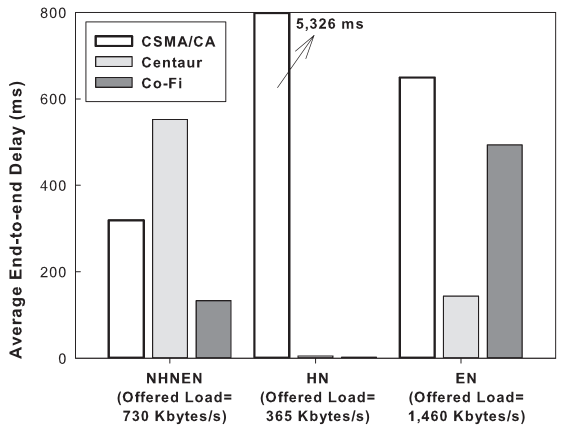


Fig. 9: Average end-to-end delay of CBR (UDP) downlink traffic for each basic scenario.

among them should not be directly compared with one another.

Under the NHNEN case, we observe that Co-Fi achieves the smallest average end-to-end delay among all schemes. Centaur performs worst because it cannot properly schedule the NHNEN traffic. In the HN case, the delays of Co-Fi and Centaur are negligible whereas CSMA/CA's delay is over 5 seconds. The low end-to-end delay of Co-Fi and Centaur is attributed to the fact that these two schemes completely avoid the HN interference through allocating non-overlapping time slots and the offered traffic rate is relatively low. The result clearly demonstrates that a centralized traffic scheduler like Co-Fi and Centaur can dramatically reduce end-to-end delays. Under the EN case, the end-to-end delay of Co-Fi is almost three times higher than Centaur's.

C. Simulation under a Realistic WLAN Environment

Configuration: We create a WLAN environment in QualNet based on the WLAN environment of one of our campus buildings as illustrated in Figure 10. The created simulation environment has three floors, and all floors have the same layout and dimension (i.e., 120m length and 40m width). In addition, we place APs at the same locations where the actual APs are located in the building whilst we distribute stations randomly within the space. Because each AP uses one of the three orthogonal channels in 2.4 GHz, we conduct our simulations under this multi-orthogonal channel environment.

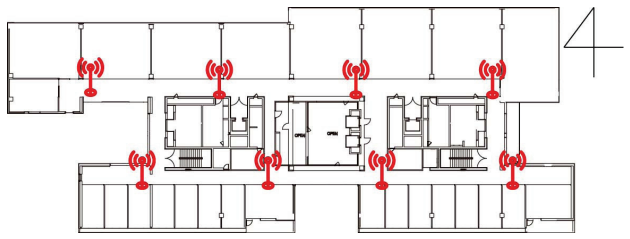


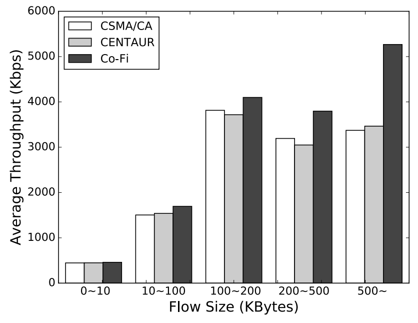
Fig. 10: Floor plan of a campus building. For simulations, stations distributed at random within the dimension of the floor plan. The characteristics of wireless medium in the simulations are set as closely to those of the building as possible. The height of a floor is around 5 meters.

We vary the number of stations (from 2 to 5 stations per AP) that are associated with each AP. Because there are 24 APs (8 APs on each floor) in total, the total number of stations varies from 48 to 120 stations, which accurately reflects WLAN environments ranging from sparse one to dense one. In addition, we use the shadowing mean of 8 dB since the wireless condition of the campus building is highly obstructed by walls and obstacles.

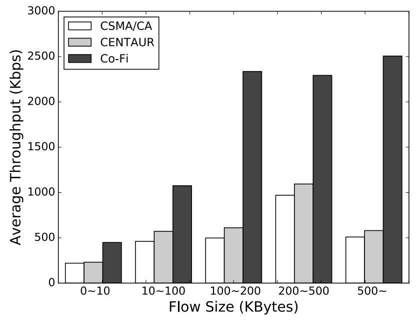
Results: We first evaluate average per-flow throughput. We cluster flows into groups based on their size and compute average throughput in each group. Figure 11 highlights that Co-Fi outperforms Centaur and CSMA/CA across all flow size groups under the three different load conditions. As expected, there is no much gain for scheduling small flows. In contrast, for TCP flows larger than 500 KB, Co-Fi achieves 3-5× higher throughput than Centaur as the load increases.

Figure 12 demonstrates cumulative distributions of averaged aggregate throughput per station. When the network is lightly loaded (i.e., 48 stations), while Co-Fi works better than Centaur and CSMA/CA; however, the amount of throughput improvement is marginal. As the network becomes more crowded (96 stations), Co-Fi experiences a slight throughput degradation compared to the 48 stations case, but Centaur and CSMA/CA face a significant performance drop as they cannot handle a high degree of NHNEN interference (2.01 times more number of NHNEN interferences than the 48 stations case). In case of 120 stations, Co-Fi still obtains the best result, but it loses almost 47% of throughput compared to its performance in the moderately-loaded case (c.f., the median throughput in Figure 12(b) is about 7.5 Mbps and the throughput in Figure 12(c) is roughly 3.5 Mbps). Thus, the throughput difference between Co-Fi and the others reduces due to the increased traffic load.

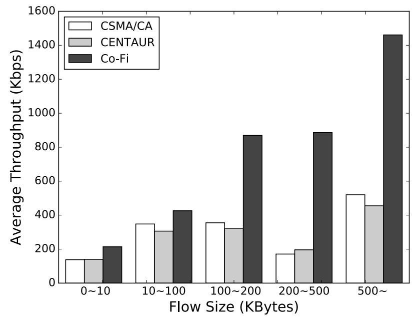
From Figures 11 and 12, we can conclude that centralized traffic scheduling mechanisms are not needed much if the network is lightly loaded. A noticeable observation is that as the network load increases, Centaur performs slightly worse than the regular CSMA/CA mechanism. This is mainly because its scheduling complexity becomes too high to bring any gain. In contrast, Co-Fi sustains such a high load because the overhead



(a) 48 stations (lightly-loaded case)

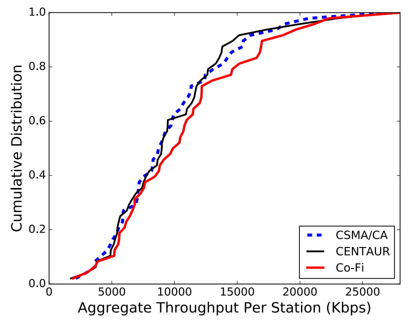


(b) 96 stations (moderately-loaded case)

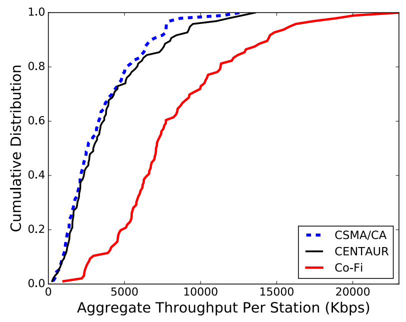


(c) 120 stations (heavily-loaded case)

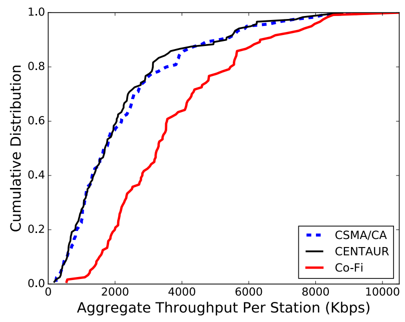
Fig. 11: Average per-flow throughput depending on different flow sizes.



(a) 48 stations (lightly-loaded case)



(b) 96 stations (moderately-loaded case)



(c) 120 stations (heavily-loaded case)

Fig. 12: Cumulative distribution of averaged aggregate throughput per station.

of enforcing traffic schedule is distributed across APs.

The results of end-to-end delay also exhibit similar trends to those of throughput. As demonstrated in Figure 13, CO-FI reduces end-to-end delay by 10% to 90%, compared to other schemes. CO-FI always obtains the smallest end-to-end delay regardless of flow sizes and total loads. In contrast, Centaur brings marginal delay gains over CSMA/CA across the simulation cases. As the network becomes denser, the level of NHNEN interference also becomes more intensive. Centaur's lack of support for NHNEN interferences blocks further delay improvement. These results showcase that the reduced end-to-end delay of CO-FI can be particularly useful for delivering real-time (multimedia) services even in densely deployed WLAN environments.

V. RELATED WORK

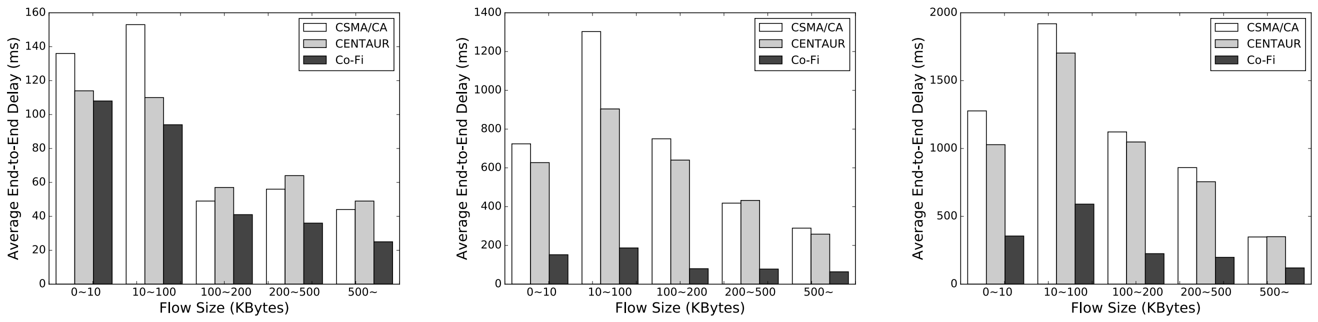
In this section, we briefly cover centralized scheduling and overlay MAC approaches that are most relevant to our work.

Centralized scheduling schemes, such as *Centaur* [6], *Shuffle* [7], and *DPS* [8], use a central controller to schedule the downlink traffic that passes through an edge router. In common, their controllers schedule every downlink frame and receives feedback from APs whenever a frame is transmitted to stations. Therefore, these schemes incur a lot of computational overhead at the controller. Further, they do not appropriately utilize the today's powerful APs because the APs strictly adhere to the transmission timing decided only by the controller. Compared to those approaches, our scheme introduces less overhead at the controller because it delegates the enforcement of time slot schedules to APs. Because Centaur is one of the most well-known centralized scheduling schemes, we did a thorough comparison of our scheme and Centaur in this work. In contrast, DOMINO [23] schedules both uplink and downlink traffic in a centralized fashion, but requires modification in PHY and MAC layers at both AP and station.

Another research topic pertaining to our scheme is overlay MAC protocols. A large body of research has focused on implementing a TDMA protocol on the 802.11-based hardware [19], [24], [25], [26]. These schemes however completely replace CSMA/CA with their own TDMA protocols. Thus, they are unfortunately incompatible with existing 802.11 devices. As the most relevant work to our hybrid MAC protocol, there exist several approaches [27], [28], [29] that implement a TDMA protocol on top of CSMA/CA without disabling it. However, these schemes have different uses of TDMA, such as fairness [27], power consumption [28], and Quality of Service (QoS) [29]. Another overlay MAC for multi-hop sensor networks switches its behavior according to the level of contention [30]. The scheme relies on a static scheduling that is determined in a distributed manner.

VI. CONCLUSION

There has been an escalated level of interference among wireless access points and stations due to dense deployment of IEEE 802.11 Wireless LANs. We introduced a coordinated Wi-Fi architecture, CO-FI, that alleviates interferences and hence boosts up wireless network performance for downlink traffic.



(a) 48 stations (lightly-loaded case) (b) 96 stations (moderately-loaded case) (c) 120 stations (heavily-loaded case)

Fig. 13: Average per-flow end-to-end delay depending on different flow sizes.

In CO-FI, a controller orchestrates access points depending on offered loads and interference types. CO-FI effectively reduces its scheduling complexity and mitigates the effect of time synchronization errors by letting a hybrid MAC protocol—*CoMAC* in the access points use CSMA/CA and TDMA protocols selectively. Our evaluation results demonstrate significantly improved throughput and end-to-end delay gain over existing approaches, especially when wireless devices are densely deployed and the networks are heavily loaded.

ACKNOWLEDGMENTS

This work was supported in part by a grant from the Royal Society.

REFERENCES

[1] A. Patro, S. Govindan, and S. Banerjee, "Observing Home Wireless Experience Through WiFi APs," in *Proceedings of ACM MobiCom*, 2013.

[2] M. A. Ergin, K. Ramachandran, and M. Gruteser, "Understanding the effect of access point density on wireless lan performance," in *Proceedings of ACM MobiCom*, 2007.

[3] V. Mhatre, K. Papagiannaki, and F. Baccelli, "Interference mitigation through power control in high density 802.11 wlans," in *Proceedings of IEEE INFOCOM*, 2007.

[4] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan, "Understanding and mitigating the impact of rf interference on 802.11 networks," in *Proceedings of ACM SIGCOMM*, 2007.

[5] Y. Lee, K. Kim, and Y. Choi, "Optimization of ap placement and channel assignment in wireless lans," in *Proceedings of IEEE LCN*, 2002.

[6] V. Shrivastava, N. Ahmed, S. Rayanchu, S. Banerjee, S. Keshav, K. Papagiannaki, and A. Mishra, "CENTAUR: Realizing the Full Potential of Centralized Wlans Through a Hybrid Data Path," in *Proceedings of ACM MobiCom*, 2009.

[7] J. Manweiler, N. Santhapuri, S. Sen, R. Choudhury, S. Nelakuditi, and K. Munagala, "Order matters: Transmission reordering in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 2, pp. 353–366, Apr. 2012.

[8] D. Zhao, M. Zhu, M. Xu, and J. Cao, "Downlink packets scheduling in enterprise wlan," in *Proceedings of IEEE WCNC*, 2013.

[9] A. Kamerman and L. Monteban, "Wavelan-ii: A high-performance wireless lan for the unlicensed band," *Bell Labs Technical Journal*, vol. 2, no. 3, pp. 118–133, 1997.

[10] M. Lacage, M. H. Manshaei, and T. Turletti, "IEEE 802.11 Rate Adaptation: A Practical Approach," in *Proceedings of ACM MSWiM*, 2004.

[11] H. Falaki, D. Lymberopoulos, R. Mahajan, S. Kandula, and D. Estrin, "A First Look at Traffic on Smartphones," in *Proceedings of ACM IMC*, 2010.

[12] A. Gupta, J. Min, and I. Rhee, "Wifox: Scaling wifi performance for large audience environments," in *Proceedings of ACM CoNEXT*, 2012.

[13] "Ericsson mobility report," http://www.ericsson.com/mobility-report, Nov. 2012.

[14] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proceedings of ACM MobiCom*, 2003.

[15] N. Ahmed and S. Keshav, "Smarta: A self-managing architecture for thin access points," in *Proceedings of ACM CoNEXT*, 2006.

[16] J. Elson and D. Estrin, "Time synchronization for wireless sensor networks," in *Proceedings of the 15th International Symposium on Parallel and Distributed Processing*, Apr. 2001, pp. 1965–1970.

[17] V. Shrivastava, S. Rayanchu, S. Banerjee, and K. Papagiannaki, "PIE in the Sky: Online Passive Interference Estimation for Enterprise WLANs," in *Proceedings of USENIX NSDI*, 2011.

[18] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," *SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[19] P. Djukic and P. Mohapatra, "Soft-TDMAC: A Software TDMA-Based MAC over Commodity 802.11 Hardware," in *Proceedings of IEEE INFOCOM*, 2009.

[20] G. Bianchi, P. Gallo, D. Garlisi, F. Giuliano, F. Gringoli, and I. Tinnirello, "MAClets: Active MAC Protocols over Hard-coded Devices," in *Proceedings of ACM CoNEXT*, 2012.

[21] "Qualnet – scalable network technologies," http://web.scalable-networks.com/.

[22] B. A. Mah, "An empirical model of http network traffic," in *Proceedings of IEEE INFOCOM*, 1997.

[23] W. Zhou, D. Li, K. Srinivasan, and P. Sinha, "DOMINO: Relative Scheduling in Enterprise Wireless LANs," in *Proceedings of ACM CoNEXT*, 2013.

[24] D. Koutsonikolas, T. Salonidis, H. Lundgren, P. LeGuyadec, Y. C. Hu, and I. Sheriff, "TDM MAC Protocol Design and Implementation for Wireless Mesh Networks," in *Proceedings of ACM CoNEXT*, 2008.

[25] W.-Z. Song, R. Huang, B. Shirazi, and R. LaHusen, "TreeMAC: Localized TDMA MAC Protocol for Real-time High-data-rate Sensor Networks," *Pervasive and Mobile Computing*, vol. 5, no. 6, pp. 750–765, Dec. 2009.

[26] C. Li, H.-B. Li, and R. Kohno, "Reservation-based dynamic tdma protocol for medical body area networks." *IEICE Transactions*, vol. 92-B, no. 2, pp. 387–395, 2009.

[27] A. Rao and I. Stoica, "An overlay mac layer for 802.11 networks," in *Proceedings of ACM MobiSys*, 2005.

[28] J. Snow, W. chi Feng, and W. chang Feng, "Implementing a low power tdma protocol over 802.11," in *Proceedings of IEEE WCNC*, 2005.

[29] J. Lee, M. Uddin, J. Tourrilhes, S. Sen, S. Banerjee, M. Arndt, K.-H. Kim, and T. Nadeem, "meSDN: Mobile Extension of SDN," in *Proceedings of the 5th International Workshop on Mobile Cloud Computing & Services*, 2014.

[30] I. Rhee, A. Warrier, M. Aia, J. Min, and M. L. Sichitiu, "Z-mac: A hybrid mac for wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 511–524, Jun. 2008.