SAP: Smart Access Point with Seamless Load Balancing Multiple Interfaces

Xi Chen, Yue Zhao, Brian Peck and Daji Qiao
Iowa State University, Ames, IA 50011
{leon6827, yuezh, bpeck, daji}@iastate.edu

Abstract—**Providing adequate Wi-Fi services to meet user demand in densely populated environments has been a fundamental challenge for Wi-Fi networks. In this paper, we explore the emerging OAMI (One-AP-Multiple-Interface) architecture and propose a unique solution called SAP (Smart Access Point). SAP takes full advantage of the OAMI architecture to provide seamless handoff experience to users, while smartly balancing the network load across multiple interfaces based on users' time-varying traffic load conditions. Moreover, we define a Traffic Fulfillment (TF) performance metric to quantify the user experience and aid in association scheduling. SAP is an AP-only solution that requires trivial network modifications and is backwards compatible with legacy 802.11 stations. We have implemented SAP in the MadWifi device driver and demonstrated its effectiveness via experiments.**

I. Introduction

Wi-Fi hotspot is one of the leading ways for individuals to connect to the Internet. Clients may use a variety of devices, including smart phones, laptops, tablets, etc., to connect to the Internet via Wi-Fi hotspots from a variety of commercial locations, including airports, hotels, retail locations, and coffee shops. As the popularity of Wi-Fi increases, the demand for bandwidth at each access point also increases. For instance, a Wi-Fi network in an airport may need to serve hundreds of users and thus a single AP would be insufficient to satisfy the network demand. One possible solution is to deploy multiple APs throughout the airport terminals so that each AP serves a fewer number of users. However, this solution may have the following issues. Firstly, in order to maximize the system performance, frequent user handoff between APs may be needed, which is a time-consuming process as a user station must disassociate from its current AP and then authenticate and reassociate with a new AP. Secondly, the APs may need to exchange information regarding the connected users to aid in association scheduling, which adds a communication overhead to the system. Finally, such a system may have issues routing downlink traffic to stations which perform a reassociation, causing packets to be delayed or lost.

An alternative solution is to equip each AP with multiple Wi-Fi interfaces operating on different channels. Such an AP architecture is called OAMI (One-AP-Multiple-Interface). In this paper, we propose a unique SAP (Smart Access Point) architecture based on OAMI, which migrates seamless the connected stations between multiple interfaces to balance the network load. Specifically, SAP has the following features:

- SAP introduces an *Interface Unification Layer* to unify the appearances of multiple interfaces, which makes seamless handoff a possibility.
- SAP utilizes a *Shared Node Table* to facilitate fast reassociation between interfaces. Reassociation can be performed simply by copying the station context to another interface and alerting the station of a BSS channel switch.
- SAP mitigates issues with downlink traffic via a *Bridge Forwarding Module* which allows packets to be forwarded to the correct interface and their final destination.
- SAP has a *Monitoring & Execution Module* which monitors various statistics for each connected station (including the traffic rate) and reports them to a *Coordination Module* to determine association scheduling.
- SAP is an AP-only solution without the need of any modifications at the stations or external routers.

The rest of the paper is organized as follows. Observations motivating SAP are described in Section II, along with accompanying design goals. The design of the SAP architecture is detailed in Section III. Section IV introduces a network model to calculate the maximum service rate of a station. We present experimental based performance evaluation results in Section V. Related work is discussed in Section VI and we conclude the paper in Section VII.

II. Motivations and Design Goals

In this section, we discuss the emerging One-AP-Multiple-Interface architecture for Wi-Fi networks and present a few interesting observations from experiments regarding the station behaviors in Wi-Fi networks, which motivate us to design and implement the proposed SAP architecture. Experiments are conducted with Dell Optiplex desktop and Latitude laptops equipped with wireless adapters which embed Atheros 5212 chipset. Each machine is loaded with a MadWifi device driver [1] to collect the experimental data.

A. Motivations and Observations

1) One-AP-Multiple-Interface Architecture: As Wi-Fi becomes more ubiquitous, in order to serve more Wi-Fi stations in a network, it is a more cost-effective solution to equip an AP with multiple interfaces operating on different channels than to deploy multiple APs in the network. However, by simply adding more interfaces to an AP without carefully planning how the interfaces shall work together to serve the stations, the OAMI (One-AP-Multiple-Interface) architecture is merely "multiple APs in the same box," which is shown in Fig. 3(a).

Innovations are needed to coordinate between multiple interfaces so stations can truly enjoy the benefits that OAMI offers over the conventional OAOI (One-AP-One-Interface) architecture. For example, as multiple interfaces reside on the same machine under OAMI, the handoff delay of moving a station from one interface to another under OAOI (which will be discussed next) could be minimized. In another example, the coordination between interfaces under OAMI could be accomplished internally without introducing extra communication overhead to the backbone network (such as Ethernet) that connects the interfaces (on different APs) under OAOI.

2) Long Handoff Delay under OAOI: The IEEE 802.11 handoff process (to move a station from an AP to another) could be very time-consuming [2], [3]. During the process, the station needs to perform channel scanning, authentication and reassociation. Moreover, additional procedures that may be needed at higher layers such as DHCP discovery could render the entire process even longer.

We examine how large the handoff delay could be with a simple experiment that uses two APs and one station. The station keeps sending Ping packets to a remote server, once every 200 ms. At the 8-second mark, the station starts the handoff process to associate with the other AP. Fig. 1 plots the RTT (Round Trip Time) of the Ping packets. As shown in the figure, the entire handoff process takes about six seconds to complete, during which the station experiences severe service disruption; this is evidenced by the drastically increased RTTs of Ping packets during the process.

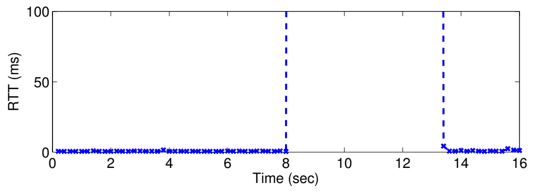


Fig. 1. Large RTT of the Ping packets during the legacy handoff process.

In practice, it is always desirable to have a seamless handoff process with minimal handoff delay and thus eliminate the service disruption, which could be critical to many user applications such as real-time video streaming.

3) Revisit of Performance Anomaly: Performance anomaly is a well-known phenomenon in 802.11 networks [4]. It is caused by the transmission rate diversity between all stations associated with the same AP. With rate diversity, the high-rate station is "slowed down" by the low-rate station because the 802.11 protocol is designed to allow all contending stations to access the channel with an equal probability. Due to throughput loss of performance anomaly, many association scheduling schemes try to avoid it by considering the transmission rates when making association decisions. For example, stations may be grouped according to their transmission rates; all the high-rate stations are associated with one AP while another AP serves all the low-rate stations.

One of the key assumptions in the performance anomaly study is that all contending stations are saturated or heavily loaded. We now revisit the performance anomaly in practical scenarios where the low-rate station may be under various loads. In the experiment, we associate two stations with the same AP; one of them communicates with the AP at the low 1 Mbps rate while the other at the high 54 Mbps rate. The high-rate station is always saturated while the load of the low-rate station increases gradually from 10 Kbps to 2000 Kbps.

As shown in Fig. 2, when the low-rate station is lightly loaded, the throughput of the high-rate station is less affected

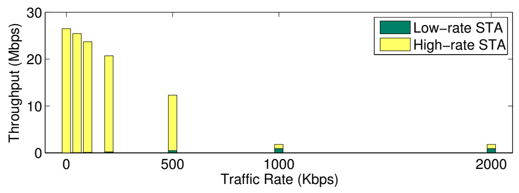


Fig. 2. Throughput comparison when the low-rate (1 Mbps) station is under different loads. Two stations are used in the experiment and the high-rate (54 Mbps) station is always saturated.

as the low-rate station only contends for the channel sporadically. As the load of the low-rate station increases, the performance anomaly becomes more apparent. When the low-rate station becomes saturated with a load of 1000 or more packets per second, the classic performance anomaly appears with two stations receiving an equal throughput and the overall system throughput suffering a significant drop.

In practice, as stations could be very diverse in terms of their traffic loads, it is important to consider the actual load condition of each station (whenever possible) when making the association scheduling decisions.

B. Design Goals

Based on the above observations, we propose in this work a SAP (Smart Access Point) realization of the OAMI architecture. With SAP, the AP shall be able to:

- adjust the association scheduling decisions between multiple interfaces dynamically based on stations' actual and varying traffic load conditions; and
- execute the association scheduling decisions with seamless handoff of stations between interfaces with minimal handoff delay, by taking full advantage of the OAMI architecture.

Moreover, SAP shall be an AP-only solution. It shall not require any modification at the client side and shall work with all legacy stations, thus facilitating its practical deployment.

III. The Proposed SAP Architecture

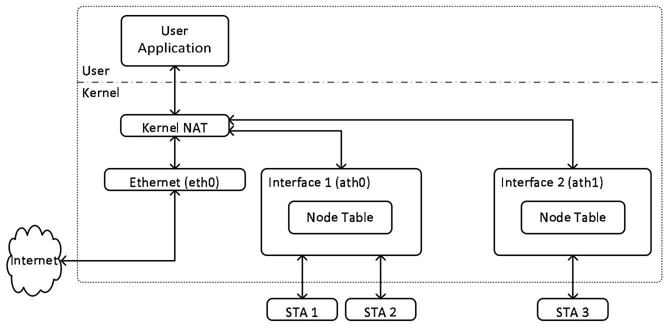
In this section, we first present an overview of the proposed SAP architecture, then describe each SAP module in detail.

A. SAP Overview

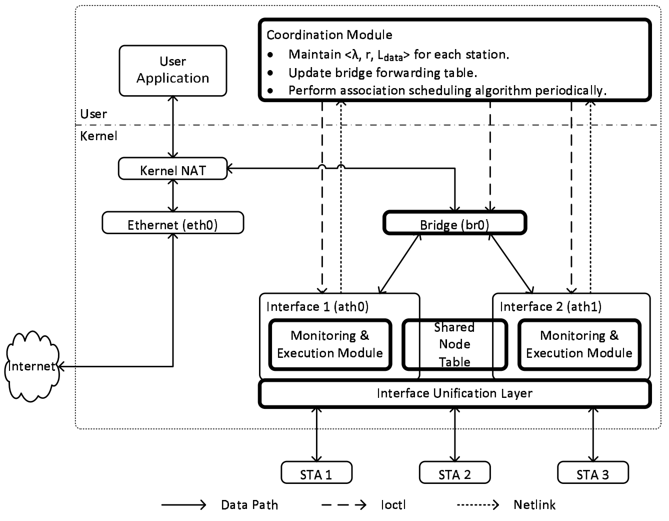
The overall structure of the proposed SAP architecture is shown in Fig. 3(b). In comparison to the basic OAMI architecture in Fig. 3(a), SAP has the following additional function blocks: (i) *Interface Unification Layer*; (ii) *Shared Node Table*; (iii) *Bridge Forwarding Module*; (iv) *Monitoring & Execution Module*; and (v) *Coordination Module*. All these modules are implemented at the AP only, without the need of any modifications at the stations or external routers.

(i) *Interface Unification Layer*. SAP creates an Interface Unification Layer to unify the appearances of multiple interfaces, which makes the handoff process transparent to the stations and thus seamless handoff a possibility. Details will be discussed in Section III-B1.

(ii) *Shared Node Table*. Different from OAMI, where each interface maintains a separate table of the stations that are associated with it, SAP allows multiple interfaces to



(a) The basic OAMI architecture.



Coordination Module
• Maintain <λ, r, L_data> for each station.
• Update bridge forwarding table.
• Perform association scheduling algorithm periodically.
User Application
User
Kernel
Kernel NAT
Ethernet (eth0)
Bridge (br0)
Interface 1 (ath0)
Monitoring & Execution Module
Shared Node Table
Interface 2 (ath1)
Monitoring & Execution Module
Interface Unification Layer
Internet
STA 1
STA 2
STA 3
Data Path
ioctl
Netlink

(b) Our proposed SAP architecture.

Fig. 3. Comparison of our proposed SAP architecture and the basic OAMI architecture. Function blocks added in SAP are marked in bold.

share a single node table. This simplifies significantly the context transfer procedure that is required by the handoff process. Details of the Shared Node Table structure will be discussed in Section III-B2.

(iii) *Bridge Forwarding Module*. In OAMI, after handoff, to make sure proper forwarding of a station's downlink traffic through the new interface to reach the station, the routing table at an external router needs to be updated. In comparison, SAP sets up an internal bridge for this purpose. Details of the Bridge Forwarding Module will be discussed in Section III-B3.

(iv) *Monitoring & Execution Module*. In SAP, each interface monitors the statistics of all the stations that are associated with it, including the physical transmission rates, the traffic rates, and the data payload sizes. Collected statistics are reported to the *Coordination Module* that uses them as inputs to its association scheduling algorithm. Meanwhile, the Monitoring & Execution Module also accepts commands from the Coordination Module and executes them by sending the corresponding management frames to the station if needed. Details of this module will be discussed in Section III-B4.

(v) *Coordination Module*. The Coordination Module in the user space coordinates the behaviors of all modules discussed above. It accepts the statistics reported by the Monitoring & Execution Modules of multiple interfaces, and runs an association scheduling algorithm periodically to determine which station should be moved to another interface. Based on the output of the algorithm, it issues commands to the Monitoring & Execution Module, and updates the Bridge Forwarding Table accordingly. Details will be discussed in Section III-B5.

B. SAP Modules

This section describes the five modules of SAP in detail.

1) Interface Unification Layer: In the basic OAMI architecture, different interfaces (i) have different MAC addresses, (ii) operate on different non-overlapping channels, and (iii) may have different BSSIDs. Therefore, in order for a station to continue receiving services from the AP via a different interface than the current one it is associated with, it needs to not only switch to the channel that the new interface operates on, but also use the BSSID and MAC address of the new interface when preparing its packets; otherwise, all its packets will simply be discarded by the new interface due to mismatch of information. For this reason, the conventional handoff process typically consists of the following steps [5]:

(i) The station performs channel scanning;
(ii) The station extracts the BSSID and MAC address of the new interface from Beacon/Probe Response frames;
(iii) The station switches to the channel that the new interface operates on;
(iv) The station exchanges management frames with the new interface to complete authentication and reassociation;
(v) The station prepares its packets using the BSSID and MAC address of the new interface and starts communicating with the AP via the new interface.

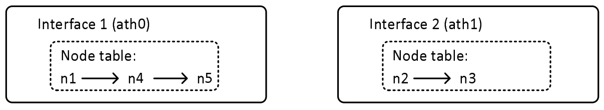
As we can see, the station is required to participate actively in the entire handoff process, where various delays occur at various steps of the process.

To simplify the handoff process and reduce the handoff delay, SAP introduces an Interface Unification Layer (IUL). It unifies all interfaces and makes them appear the same (except that they operate on different non-overlapping channels) to the stations by setting the same BSSID and the same MAC address for all interfaces. This could be done via, for example, the `ioctl` command (to set the BSSID) and the `ath_attach()` function (to set the MAC address) in MadWifi. As a result, Step (v) of the handoff process could be avoided completely. As we will discuss next, with the help from the Shared Node Table structure and the Bridge Forwarding Module, which help remove Step (iv), and the Monitoring & Execution Module together, which help remove Steps (i) and (ii), the handoff process is simplified to a channel switch operation, thus reducing the handoff delay to the minimum.

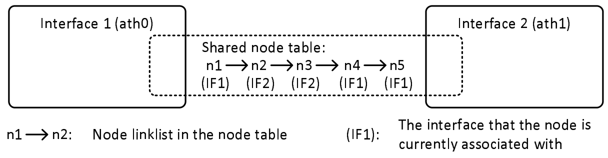
2) Shared Node Table: As we mentioned in Section III-B1, in the basic OAMI architecture, a station has to perform authentication and reassociation before being served again by the new interface. The root reason for this requirement is that, under OAMI, each interface maintains a separate node table that keeps track of only the stations that are associated with it, as shown in Fig. 4(a). Therefore, the context transfer procedure during the handoff process under OAMI consists of the following steps:

- Firstly, the station disassociates from the current interface which then removes the corresponding entry from its node table;
- Then, the station scans, authenticates (e.g., key exchanges) and reassociates with the new interface;
- Finally, the new interface creates a new entry in its node table, which is populated by the context information of the station, such as security information, association information, and so on.

As we can see, the station has to go through the time-consuming authentication and reassociation in order to create an entry in the new interface's node table.



(a) Default node table structure.



Interface 1 (ath0)
Interface 2 (ath1)
Shared node table:
n1 ⟶ n2 ⟶ n3 ⟶ n4 ⟶ n5
(IF1) (IF2) (IF2) (IF1) (IF1)
n1 ⟶ n2: Node linklist in the node table
(IF1): The interface that the node is currently associated with

(b) Shared node table structure in SAP

Fig. 4. Comparison of the shared node table structure in SAP and the default node table structure.

The context transfer procedure described above does not take advantage of the fact that the interfaces reside on the same machine under OAMI, which means that they are served by the same kernel and the same memory space. Hence, it is possible to simplify the context transfer procedure via internal actions, which is exactly what SAP does. Specifically, SAP maintains a single node table shared by all interfaces, as shown in Fig. 4(b). Each entry in the shared node table now has an extra flag to indicate the interface that the station is currently associated with. With such a shared node table structure, the context transfer procedure is simplified to a simple modification of a flag in the shared node table. As a result, authentication and reassociation are avoided completely.

3) Bridge Forwarding Module: In the basic OAMI architecture, upon a successful handoff of a station to a new interface, the external router needs to update its routing table so the downlink packets for the station could be routed properly toward the new interface. SAP uses a different forwarding mechanism to achieve this purpose. It is an internal approach that limits all the modifications within the AP only.

Specifically, SAP establishes an internal bridge that interconnects the multiple interfaces at the data link layer, and each interface is assigned a bridge *port* number. The bridge has a *forwarding table* which contains entries for each station that is associated with the AP. Each entry contains the following information: (i) the MAC address of the station; and (ii) the bridge port number of the interface that the station is associated with. As shown in Fig. 3(b), all the downlink packets are first routed to the bridge, and then forwarded to the appropriate interface by looking up the bridge forwarding table. Upon a successful handoff of a station, SAP updates the corresponding entry in the bridge forwarding table, in addition to the corresponding entry in the Shared Node Table discussed in Section III-B2.

4) Monitoring & Execution Module: In SAP, each interface monitors the stations that are associated with it and collects the station statistics for facilitating the association scheduling algorithm in the Coordination Module. The collected statistics include the downlink and uplink traffic rates, the transmission rate and the data payload length of each station. SAP maintains a moving average for each statistic, and updates it upon each packet transmission/reception.

Based on the collected statistics, each interface reports the following information to the Coordination Module periodically: λ, r and L_{data}. Here, λ is the overall traffic rate of the station, and we use the uplink throughput to approximate the uplink traffic rate of the station. r is the communication rate between the station and the AP. L_{data} is the data payload length of the station. In our implementation of SAP in the MadWifi device driver, we create `netlink` sockets to enable the information exchange between the Monitoring & Execution Module (in the system kernel) and the Coordination Module (in the user space).

Another function of the Monitoring & Execution Module is that it accepts an `ioctl` command from the Coordination Module and executes it by sending a corresponding management frame to the station. For example, if the Coordination Module determines that the station with a MAC address of `00:1c:f0:f9:a3:4c` should be moved from Interface `ath0` that the station is currently associated with to a different interface that operates on Channel 11, it issues the following `ioctl` command:

```
iwpriv ath0 switch 00:1c:f0:f9:a3:4c 11
```

to Interface `ath0`. Subsequently, the Monitoring & Execution Module of Interface `ath0` sends an `Channel Switch Announcement (CSA)` management frame to notify the station to switch to Channel 11. Note that `Channel Switch Announcement (CSA)` is a standard management frame defined in the 802.11h [6] standard and its original purpose was to allow an AP to notify all its associated stations of the BSS channel change after it changes its own operating channel. Once the station receives the management frame, it will follow the standard to switch to the new channel. The creative use of the `Channel Switch Announcement (CSA)` management frame is another key factor that contributes to SAP's seamless handoff process.

Seamless Handoff in SAP So far, we have discussed the Interface Unification Layer, the Shared Node Table structure, the Bridge Forwarding Module and the Monitoring & Execution Module, which are the four key SAP components that work together to enable seamless handoff. To summarize, with SAP, the handoff of a station between interfaces is simplified to

- two simple internal AP actions to update the Shared Node Table (i.e., modification of a flag) and the Bridge Forwarding Table (i.e., modification of an entry); and

- a simple `ioctl` command from the AP to the old interface which then sends a management frame to notify the station of the channel switch.

Comparing with the conventional handoff process that consists of channel scanning, channel switch, station authentication, station reassociation and the accompanying upper layer actions such as DHCP discovery, the handoff delay in SAP has been reduced to the minimum: the channel switch time.

5) Coordination Module: SAP's Coordination Module resides in the user space. It collects the statistics reported by the Monitoring & Execution Modules of multiple interfaces, and uses them as inputs to an association scheduling algorithm. The goal of the algorithm is to find the *association pattern* – which specifies the station-to-interface association for all stations – that provides the best user experience to all stations.

In the past, many performance metrics have been proposed to quantify the user experience, such as throughput, airtime, and so on. In this paper, we define a performance metric called *traffic fulfillment* (TF) and use the minimum traffic fulfillment among all stations as a quantitative measure of the user experience. Note that SAP is not limited or bound to this particular metric and can work fine with other performance metrics as well by adjusting the association scheduling algorithm accordingly. For a given association pattern, *traffic fulfillment* (TF) of a station is defined as:

$$\text{TF} = \frac{G}{\min\{r, \lambda\}}, \tag{1}$$

where G is the actual throughput obtained by the station, r is the transmission rate between the station and the AP, and λ is the traffic rate required by the station. TF is defined this way to reflect how well the station's requested traffic rate can be fulfilled by the system, which could be a good measurement of the user experience. Clearly, different association patterns may result in different G. We will discuss how to estimate G in Section IV.

Formally, the optimization problem that the association scheduling algorithm tries to solve is defined as follows. Given the transmission rates (r), the traffic rates (λ), and the data payload lengths (L_{data}) of all stations, the goal is to find the best association pattern that maxi-minimizes the traffic fulfillment (TF) among all stations:

$$\text{pattern}^{\text{OPT}} = \arg \max_{\{\text{all association patterns}\}} \text{TF}_{\min}, \tag{2}$$

where $\text{TF}_{\min} = \min_i \text{TF}_i$ is the minimum TF among all n stations given an association pattern. Note that, under certain circumstances, there might exist multiple association patterns that all produce a high level of traffic fulfillment to all stations, e.g., $\text{TF}_{\min} \geqslant 0.9$. We call these association patterns *candidate patterns*. A secondary goal of the algorithm is to choose from the candidate patterns the one that maxi-minimizes the *maximum service rate* (MSR) among all stations:

$$\text{pattern}^{\text{OPT}} = \arg \max_{\{\text{all candidate patterns}\}} \text{MSR}_{\min}, \tag{3}$$

where $\text{MSR}_{\min} = \min_i \text{MSR}_i$. Here, the *maximum service rate* (MSR) is defined as the maximum throughput that a station may obtain if it increases its traffic rate to infinity, given that the status remains unchanged for all other stations competing with this station for the same interface. The secondary goal is defined this way to ensure that SAP selects the candidate pattern that is most capable of handling a sudden increase of traffic rate requirement from an arbitrary station. We will discuss how to calculate MSR in Section IV.

The optimization problems in Eqs. (2) and (3) have been proved to be NP-hard [7]. Many approximation algorithms have been proposed in the literature to solve similar problems. As the main focus of this paper is not to design a better approximation algorithm, we propose a simple and practical greedy solution to be used in the association scheduling algorithm; evaluation results in Section V show that great performance gain can be achieved even with such a simple solution. The idea is that, instead of searching over all possible association patterns (which grows exponentially with n – the total number of stations in the network), the association scheduling algorithm switches at most one station to a different interface during each scheduling period. This way, the search space is effectively reduced to $n(m-1)$ where m is the number of interfaces, hence improving the scalability of the algorithm significantly. Algorithm 1 gives the pseudo-code of the proposed solution for an AP with two interfaces. As shown in the pseudo-code, during each scheduling period, the algorithm determines whether or which station should be switched to the other interface so that $\text{TF}_{\min}$ can be increased the most. If the algorithm finds one successfully, the Coordination Module issues an `ioctl` command to the corresponding interface, which in turn notifies the station to make the channel switch, as already discussed in Section III-B4.

Algorithm 1 The Association Scheduling Algorithm in SAP

```
1:  INPUT: pattern^curr: $\vec{t_1}$ associated with IF$_1$, $\vec{t_2}$ associated with IF$_2$
2:  OUTPUT: station $x^*$ that should be switched to the other interface
3:  /* Initialization*/
4:  $\text{TF}_{\text{maximin}} = \text{TF}_{\min}$ of the current association pattern
5:  $\text{MSR}_{\text{maximin}} = \text{MSR}_{\min}$ of the current association pattern
6:  $\text{pattern}^{\text{TF}} = \text{pattern}^{\text{curr}}$; $\text{pattern}^{\text{MSR}} = \text{pattern}^{\text{curr}}$
7:  for each $x$ in $\vec{t_1} \cup \vec{t_2}$ do
8:      Consider the new association pattern by switching $x$
9:      Calculate $\text{TF}_{\min}$ and $\text{MSR}_{\min}$
10:     if $\text{TF}_{\min} > 1.1 * \text{TF}_{\text{maximin}}$ then
11:         $\text{TF}_{\text{maximin}} = \text{TF}_{\min}$; record pattern to $\text{pattern}^{\text{TF}}$
12:     end if
13:     if $\text{TF}_{\min} \geqslant 0.9$ then
14:         if $\text{MSR}_{\min} > 1.1 * \text{MSR}_{\text{maximin}}$ then
15:             $\text{MSR}_{\text{maximin}} = \text{MSR}_{\min}$
16:             record pattern to $\text{pattern}^{\text{MSR}}$
17:         end if
18:     end if
19: end for
20: if $\text{TF}_{\text{maximin}} \geqslant 0.9$ then
21:     $x^* = $ set-diff($\text{pattern}^{\text{MSR}}$, $\text{pattern}^{\text{curr}}$)
22: else
23:     $x^* = $ set-diff($\text{pattern}^{\text{TF}}$, $\text{pattern}^{\text{curr}}$)
24: end if
```

IV. Modeling and Analysis

In this section, we present a simple model to calculate the maximum service rate (MSR) of a station.

A. Calculation of MSR with Two Contending Stations

We first consider the basic scenario when two and only two stations (s and t) are associated with the same interface and contend for channel access. The transmission rate, the traffic rate and the data payload length of a station are denoted as r, λ and L_{data}, respectively. By definition, the MSR of station s is the maximum throughput that s may obtain when it increases λ_s to ∞ (i.e., becomes saturated), given that the traffic rate of station t (λ_t) remains unchanged.

Similar to [8]–[10], we use *virtual slot* to characterize the airtime usage, which is defined as a full transmission cycle including backoff, data transmission time and ACK transmission time, or a single slot time if no station is contending for the channel. Let q denote the probability that a station contends for the channel at the end of a virtual slot. Clearly, when calculating the MSR of station s, q_s is always one as s is assumed to be saturated. Therefore, we have approximately the following:

$$\begin{aligned} P_s &= q_s(1-q_t) + \frac{1}{2}q_s q_t = 1 - \frac{1}{2}q_t, \\ P_t &= q_t(1-q_s) + \frac{1}{2}q_s q_t = \frac{1}{2}q_t, \end{aligned} \tag{4}$$

where P_s and P_t are the probability that s and t completes a transmission successfully in a virtual slot. Eq. (4) is based on the assumption that two stations have an equal probability of 0.5 to seize the channel successfully when they both contend in the same virtual slot. We derive q_t based on a Markov chain model we have proposed to characterize the behaviors of station t. It has $N+1$ states where N is the maximum queue length allowed at t. State i ($0 \leqslant i \leqslant N$) of the Markov chain represents that t's queue length is $Q_t = i$ at the end of a virtual slot. Thus, by definition, we have $q_t = P(Q_t \neq 0) = 1 - \pi_0$ where π_0 is the steady state probability of state 0. Plugging q_t back to Eq. (4), P_s and P_t can then be derived. Detailed explanation of state transition probabilities and calculation of steady state probabilities are omitted due to space limitation (see [11] for details). With P_s and P_t, the MSR of station s can be calculated with Algorithm 2.

Algorithm 2 Calculation of MSR of s given that only a single station t contends with s.

```
1: INPUT: $s$, $t$ and their status
2: OUTPUT: MSR$(s,t)$
3: Calculate $T_s$ and $T_t$ based on $r_s$, $L_{\text{data}}(s)$, $r_t$, $L_{\text{data}}(s)$
4: Calculate $P_s$ and $P_t$ based on the Markov chain model
5: $T_{\text{avg}} = P_s T_s + P_t T_t$
6: $\text{MSR}(s,t) = P_s L_{\text{data}}(s)/T_{\text{avg}}$
```

B. Calculation of MSR with Multiple Contending Stations

Algorithm 3 estimates the MSR of station s given that a set of stations $\vec{\mathbf{t}}$ compete with s for the same interface. The basic idea is to pair s with the first station in $\vec{\mathbf{t}}$, say x, and treat s and x together as a virtual node with an updated effective transmission duration. Then the virtual node is paired with the second station in $\vec{\mathbf{t}}$. The process goes on until all stations in $\vec{\mathbf{t}}$ have been considered. As we will show next, this simple approximation performs reasonably well in various scenarios.

Algorithm 3 Calculation of MSR$_s$ given that a set of stations $\vec{\mathbf{t}}$ contend with s.

```
1: INPUT: $s, \vec{\mathbf{t}}$ and their status
2: OUTPUT: MSR$(s, \vec{\mathbf{t}})$
3: for each $x$ in $\vec{\mathbf{t}}$ do
4:     Run Algorithm 2 to calculate MSR$_{\text{tmp}}$ = MSR$(s,x)$
5:     // Treat $s$ and $x$ as a virtual node with an updated $T_s$
6:     Update $T_s = L_{\text{data}}(s)/\text{MSR}(s,x)$
7:     Update $\vec{\mathbf{t}} = \vec{\mathbf{t}} \setminus \{x\}$
8: end for
9: MSR$(s, \vec{\mathbf{t}}) = \text{MSR}_{\text{tmp}}$
```

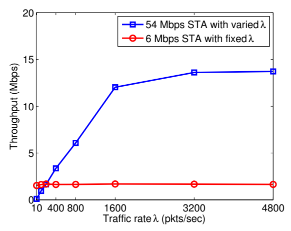
C. Validation of the Model and Estimation of G

We use ns-2 [12] simulation results to validate the above model and algorithms for calculating the MSR. In the first simulation, there are two stations in the network. STA$_1$ has a PHY rate of 54 Mbps and STA$_2$ has a PHY rate of 6 Mbps. We fix the traffic rate of STA$_2$ to be 200 packets/s, where each packet carries 1500 bytes of data. We increase the traffic rate of STA$_1$ gradually. The throughput of two stations are plotted in Fig. 5(a). In this figure, we observe that the throughput of STA$_1$ first increases almost linearly with the traffic rate, and then keeps almost constant around 13.7 Mbps when the traffic rate is high. In other words, the MSR of STA$_1$ is 13.7 Mbps. Based on this observation, we approximate the throughput of a station as follows:

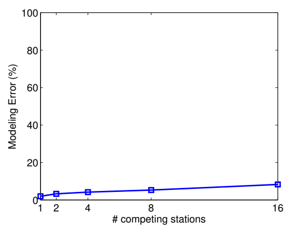
$$G = \min\{\text{MSR}, \lambda\}. \tag{5}$$

In comparison, our model gives an MSR of 13.83 Mbps, which is very close to the simulation result.

In the second simulation, we compare the simulated and analytical MSR results in a random setup. In this setup, we vary the number of competing stations, and each station chooses its PHY rate and traffic rate randomly. We plot the difference between the simulated and analyzed MSR results in Fig. 5(b). It shows that with a single competing station, our modeling scheme has an estimation error of about 2%. With 16 competing stations in the network, our model yields an estimation error of around 8.7% which is an acceptable tradeoff considering the simplicity of our model and low

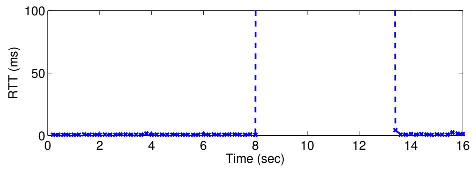


(a) Simulation results of two contending stations.

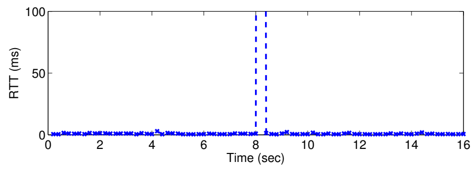


(b) Difference between simulated and analytical MSR results.

Fig. 5. Simulation-based validation of the model and algorithms for calculating the MSR.



(a) Two OAOI access points.



(b) SAP access point with two interfaces.

Fig. 6. Comparison of handoff delay. Handoff starts at around the 8-second mark. Large round trip time indicates packet loss.

computational complexity of the algorithms. The estimation error is due mainly to the approximation used in the algorithms and that the station backoff is not considered in the model.

V. Experimental Study

We have implemented SAP in MadWifi [1]. In this section, we evaluate its effectiveness using experimental results.

A. *Experiment Setup*

All the experiments are conducted with Dell desktops and laptops equipped various WLAN adapters, which all embed Atheros 5212 chipsets. The AP uses a Dell Optiplex GX520 desktop with two NetGear WAG311 PCI adapters. Clients use a Dell Latitude laptop equipped with one D-Link WNA1330 PCMCIA adapter. We use the off-the-shelf hardware instead of sophisticated equipments to conduct experiments as this makes our experimental results comparable to what users of commodity 802.11 devices may expect in realistic scenarios. We use Iperf [13] as the UDP packet generator to generate time-varying traffic loads. The association scheduling algorithm runs every 15 seconds in the SAP configuration. Statistics are reported to the Coordination Module every 5 seconds.

We compare the performance of SAP against the following schemes: (i) the conventional OAOI scheme in which each AP only has a single interface; and (ii) a variant of the SAP scheme called SAP-ST which assumes all stations are saturated in the association scheduling algorithm, i.e., without considering the station's traffic load [2], [7], [14]. Note that, by placing two OAOI access points together and have them operate on different non-overlapping channels, it is equivalent to a basic OAMI access point with two interfaces. In this section, we first perform controlled experiments to evaluate the delay performance as well as the effectiveness of the load balancing algorithm. Then, we evaluate SAP in a random setup with more stations and more realistic traffic patterns.

B. *Handoff Delay Performance*

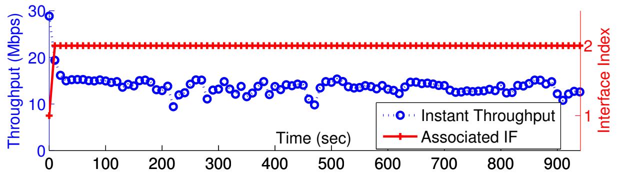
We first evaluate the delay performance of SAP with its seamless handoff architecture. In this experiment, we have a static SAP with two interfaces and one static station. The station keeps generating Ping packets (at an interval of 200 ms), which provides straightforward RTT measurements. At the 8-second mark, SAP switches the station to the other interface. Fig. 6(b) plots the RTT of the Ping packets. Recall that we have performed the same experiment under OAOI in Section II-A2; we re-plot the results in Fig. 6(a). As shown in the figures, with SAP, the entire handoff delay is around 200 ms (for the station to perform the channel switch), which is significantly smaller than the six-second delay under OAOI. Note that, the station experiences one packet loss with SAP due to non-negligible channel switch time.

C. *Load Balancing Performance*

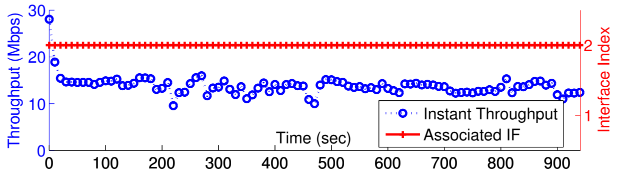
We then evaluate the load balancing performance of SAP with its smart association scheduling algorithm. In this experiment, we have one AP and three stations in the network. We manually set the PHY rate of each station as follows. Both STA_1 and STA_2 have a PHY rate of 48 Mbps while STA_3 has a PHY rate of 6 Mbps. All stations are receiving dowlink traffic with various traffic loads. STA_1 and STA_2 are heavily loaded ($\lambda \approx 30$ Mbps), while STA_3 has the following time-varying traffic pattern: $\lambda \approx 200$ Kbps between 0 and 300 seconds, 6 Mbps between 300 and 600 seconds, and 200 Kbps thereafter. STA_1 and STA_3 are initially associated with Interface 1, while STA_2 is associated with Interface 2 at the beginning. We collect the instant throughput (reported every 10 seconds), associated interface index and the TF value of each station, as well as the system aggregate throughput over run time. We compare SAP against SAP-ST in this experiment.

Performance of SAP-ST is shown in Fig. 7. SAP-ST uses saturation analysis without considering users' current traffic loads. Hence, SAP-ST schedules station-to-AP associations based only on the PHY rate of each station. As we can see in Figs. 7(a) (b) and (c), all three stations stay with the same interface throughout the evaluation period, even when STA_3 suddenly increases its traffic rate around the 300-second mark. With the saturation analysis, SAP-ST separates high-rate stations (i.e., STA_1 and STA_2 associated with Interface 2) from the low-rate station (i.e., STA_3 associated with Interface 1), assuming that the performance anomaly caused by STA_3 always exists. With this association pattern, Interface 2 serves two heavily-loaded stations, but Interface 1 serves one station that is lightly-loaded for most of the run time, thus the loads on two interfaces are largely unbalanced. The aggregate system throughput and TF values are shown in Fig. 7(d). As we can see, SAP-ST actually satisfies STA_3 (i.e., $TF_3 \approx 1$) but sacrifices STA_1 and STA_2 (i.e., $TF_{1,2} \approx 0.5$). As a result, the total system throughput is only about 30 Mbps.

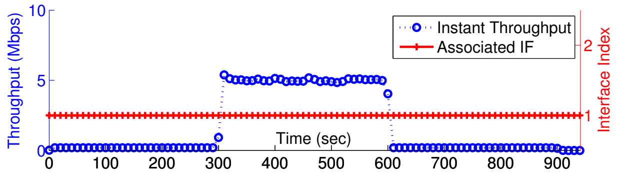
In comparison, the performance of SAP is shown in Fig. 8. By considering stations' traffic loads, SAP adjusts the association pattern dynamically. As we can see, one of the high-rate stations (i.e., STA_1) is associated with Interface 1 (same as STA_3) at the beginning when STA_3 is lightly-loaded hence no performance anomaly is observed. During this period, both STA_1 and STA_3 can be well satisfied and obtain a throughout close to the traffic rate requested. At the 300-second mark, STA_3 increases its traffic rate, due to which SAP smartly switches STA_1 to Interface 2 (same as STA_2) in order to achieve better satisfaction level for all stations. After STA_3 resumes its low traffic rate at the 600-second mark, SAP switches STA_1 back to Interface 1. In Fig. 8(d), we clearly see that the system throughout improves to around 60 Mbps for



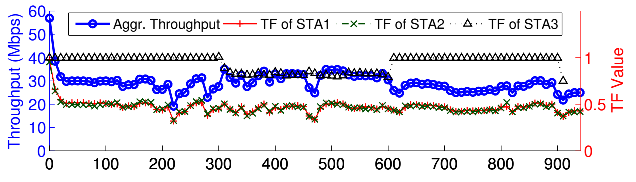
(a) Instant throughput and associated interface index for STA$_1$.



(b) Instant throughput and associated interface index for STA$_2$.

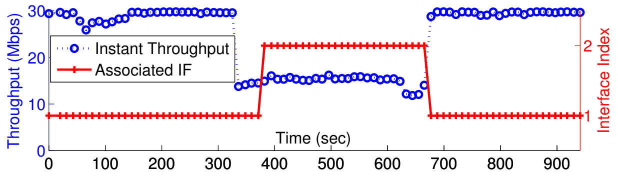


(c) Instant throughput and associated interface index for STA$_3$.

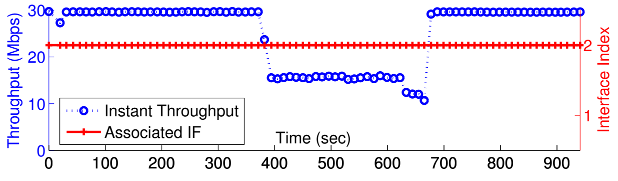


(d) Aggregate throughput and traffic fulfillment values.

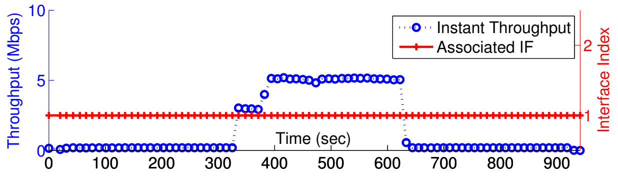
Fig. 7. Throughput and fairness performances of SAP-ST. STA$_1$ and STA$_2$ have around 30 Mbps traffic rate. STA$_3$ has time varying traffic pattern: about 200 Kbps in [0 300) sec, 6 Mbps in [300 600) sec and 200 Kbps thereafter.



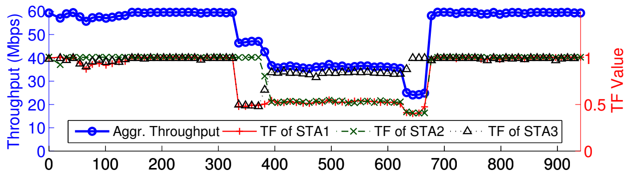
(a) Instant throughput and associated interface index for STA$_1$.



(b) Instant throughput and associated interface index for STA$_2$.



(c) Instant throughput and associated interface index for STA$_3$.



(d) Aggregate throughput and traffic fulfillment values.

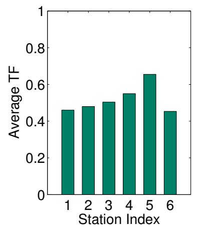
Fig. 8. Throughput and fairness performances of SAP under the same station setup as in Fig. 7.

two thirds of the evaluation period, during which all stations are well served (i.e., TF$_{1,2,3} \approx 1$).

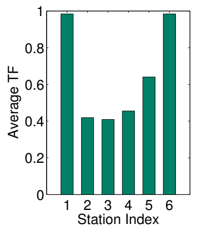
D. More Realistic Scenario

Lastly, we evaluate the performance of SAP in a more realistic scenario with six stations in the network, namely STA$_{1\ldots 6}$. In our setup, we positions six stations at different locations in our department building such that $r_4 \approx r_5 \approx 6$ Mbps, $r_2 \approx r_3 \approx 12$ Mbps, $r_6 \approx 24$ Mbps and $r_1 \approx 48$ Mbps. We uses Iperf to generate random downlink traffic to each station as follows. After every 75 seconds (on average) of idle time, Iperf randomly starts one of the following five traffic sessions: (i) Web browsing (30 seconds, about 30 Kbps bandwidth requirement); (ii) Audio (60 seconds, $\sim$100 Kbps); (iii) Video (10 minutes, $\sim$500 Kbps); (iv) HD Video (10 minutes, $\sim$6 Mbps) and (v) FTP (4 minutes, $\sim$20 Mbps). We run the experiments for one hour and record the TF value of each station during the entire experiment. We compare SAP against SAP-ST and a single OAOI access point in this scenario. Results are shown in Fig. 9.

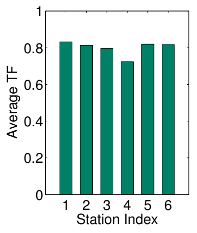
Figs. 9(a), (b) and (c) plot the average TF value of each station during the experiment. From Figs. 9(a) and (b), we observe that without carefully coordinating the multiple interfaces, four out of six stations receive similar services in SAP-ST and OAOI, even though the number of interfaces has doubled in SAP-ST. Moreover, as shown in Fig. 9(b), SAP-ST favors high-rate stations (i.e., STA$_1$ and STA$_6$) which achieves a much larger TF value ($\sim$1.0) than low-rate stations



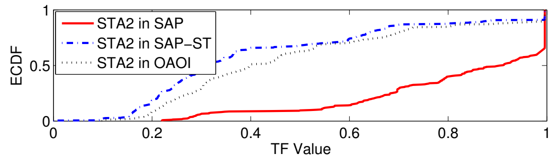
(a) A single OAOI access point.



(b) SAP-ST.



(c) SAP.



(d) ECDF of TF values of STA$_2$.

Fig. 9. Performance comparison in a more realistic scenario with six stations with various PHY rates and time-varying traffic rates.

($\sim$0.4). The reason is that SAP-ST always discriminates low-rate stations due to performance anomaly, thus it groups high-rate stations and low-rate stations to different interfaces. As a result, high-rate stations can be well served by one interface while low-rate ones are crowded at another interface. In comparison, Fig. 9(c) shows that the TF values in SAP are more balanced than SAP-ST and much higher than OAOI. The minimum TF value in SAP is 0.72, in comparison to 0.41 in SAP-ST and 0.45 in OAOI. As an example to see how SAP provides better traffic fulfillment services to stations, we plot the ECDF of the TF values of STA$_2$ in Fig. 9(d). From the figure, we can see that STA_2 in SAP has TF $\geqslant 0.5$ for more than 90% of the time, while the portion is only 32% in SAP-ST and 37% in OAOI respectively.

VI. Related Work

Related work in the literature regarding association control in IEEE 802.11 networks can be classified into two main categories, namely *handoff delay optimization* and *association metric design*.

1) Handoff delay optimization: Various fast handoff schemes have been proposed in the past with the purpose of reducing various components of handoff delay. In [3], the authors showed that the scan phase is the most significant contributor to the overall handoff delay and the variations in channel dwell time account for the large variations in the handoff delay. The schemes proposed in [15], [16] try to reduce the number of channels to probe during the scan phase. For example, [16] adopts a neighbor graph approach and the station only probes the channels that have active APs operating on. D-Scan [17], Proactive-Scan [18], [16], and [19] try to reduce channel dwell time. SyncScan [20] reduces the probe duration by having all APs synchronize their Beacon frames in the case of passive scan. D-Scan, Proactive-Scan and the scheme in [21] interleave the long scan phase to reduce the packet delay. Different from the above schemes which all try to reduce the scan delay, 802.11r [22] and the scheme in [23] aim to reduce the authentication and reassociation delays. HaND [2] removes the channel dwell time in scan delay via letting AP make the handoff decisions for stations. Compared with the previous works, SAP eliminates the scan delay, the authentication delay and the reassociation delay entirely, thus reducing the handoff delay to the minimum.

2) Association metric design: Another research topic in this area is how to decide the station-to-AP association pattern. One of the commonly-used metrics is the signal strength. For example, hysteresis-based approaches in [17], [18], [21] direct a station to reassociate with the new AP that has a stronger signal strength. [15] compares several association metrics that are based on signal strength statistics. [24] combines the signal strength value with the large-time-scale performance measurement. In [25], another metric is proposed which is called available capacity. [2], [7], [9], [14], [26] consider the fairness among all stations in the association control. Specifically, [2], [7], [14] propose a throughput fulfillment factor. [26] provides air time fairness among stations. By considering the traffic rate instead of saturation analysis, [9] adopts a proportional fairness metric taking the station's traffic need into account, and evaluate it via simulation only. In comparison, SAP proposes a new association metric by considering stations' current traffic rates, and evaluates it with real-world experiments.

VII. Conclusion

From experiments, we observe that the handoff delay with legacy Wi-Fi APs is significant, and the severity of the performance anomaly in Wi-Fi networks varies with the traffic rates of connected clients. Based on these observations, we propose an advanced One-AP-Multiple-Interface (OAMI) architecture called SAP (Smart Access Point) to perform association scheduling from a unified AP and provide seamless handoff experience to users. Additionally, we introduce a Traffic Fulfillment (TF) performance metric to aid in grouping stations during association scheduling. We have implemented SAP in the MadWifi device driver and demonstrated its effectiveness using experimental results.

References

[1] Multiband Atheros Driver for Wifi, http://www.madwifi-project.org/.
[2] X. Chen and D. Qiao, "HaND: Fast handoff with null dwell time for ieee 802.11 networks," in *IEEE InfoCom'10*, 2010.
[3] A. Mishra, M. Shin, and W. Arbaugh, "An empirical analysis of the ieee 802.11 mac layer handoff process," in *ACM Computer Communications Review*, vol. 33, no. 2, 2003.
[4] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," in *IEEE InfoCom'03*, 2003.
[5] "IEEE Standard 802.11-2007, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, (Revision of IEEE Std 802.11-1999)," June 2006.
[6] "IEEE Standard 802.11h-2003: Spectrum and Transmit Power Management Extensions in the 5 GHz Band in Europe." December 2003.
[7] Y. Bejerano, S. J. Han, and L. E. Li, "Fairness and load balancing in wireless LANs using association control," in *IEEE/ACM Trans. Networking*, June 2007.
[8] D. Malone, K. Duffy, and D. J. Leith, "Modeling the 802.11 distributed coordication function in non-saturated heterogeneous conditions," in *IEEE/ACM Trans. Networking*, Feb. 2007.
[9] M. Laddomada, F. Mesiti, M. Mondin, and F. Daneshgaran, "On the throughput performance of multirate ieee 802.11 networks with variable-loaded stations: Analysis, modeling, and a novel proportional fairness criterion," in *IEEE Trans. Wireless Communications*, May 2010.
[10] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," in *IEEE JSAC*, Mar. 2000.
[11] Tech. Rep., http://home.eng.iastate.edu/~daji/papers/SAP-tech-rep.pdf.
[12] The Network Simulator - ns-2, http://http://www.isi.edu/nsnam/ns/.
[13] Iperf, http://dast.nlanr.net/projects/Iperf.
[14] W. Zhou and D. Qiao, "Fulfillment-based fairness: A new fairness notion for multi-AP wireless hotspots," in *IEEE ICC'07*, 2007.
[15] V. Mhatre and K. Papagiannaki, "Using smart triggers for improved user performance in 802.11 wireless networks," in *ACM Mobisys'06*, 2006.
[16] M. Shin, A. Mishra, and W. Arbaugh, "Improving the latency of 802.11 hand-offs using neighbor graphs," in *ACM Mobisys'04*, 2004.
[17] J. Teng, C. Xu, W. Jia, and D. Xuan, "D-scan: Enabling fast and smooth handoffs in AP-dense 802.11 wireless networks," in *IEEE InfoCom Miniconference'09*, 2009.
[18] H. Wu, K. Tan, Y. Zhang, and Q. Zhang, "Proactive scan: Fast handoff with smart triggers for 802.11 wireless lan," in *IEEE InfoCom'07*, 2007.
[19] H. Velayos and G. Karlsson, "Techniques to reduce the ieee 802.11b handoff time," in *IEEE ICC'04*, 2004.
[20] I. Ramani and S. Savage, "Syncscan: Practical fast handoff for 802.11 infrastructure networks," in *IEEE InfoCom'05*, 2005.
[21] Y. Liao and L. Gao, "Practical schemes for smooth mac layer handoff in 802.11 wireless networks," in *IEEE WoWMoM'06*, 2007.
[22] "IEEE Standard 802.11r-2008: Fast Basic Service Set (BSS) Transition." July 2008.
[23] A. Mishra, M. Shin, and W. Arbaugh, "Context caching using neighbor graphs for fast handoffs in a wireless network," in *IEEE InfoCom*, 2004.
[24] A. Giannoulis, M. Fiore, and E. Knightly, "Supporting vehicular mobility in urban multi-hop wireless netwroks," in *ACM MobiSys'08*, 2008.
[25] R. Murty, J. Padhye, R. Chandra, A. Wolman, and B. Zill, "Designing high performance enterprise wi-fi networks," in *USENIX NSDI'08*, 2008.
[26] T. Joshi, A. Mukherjee, Y. Yoo, and D. P. Agrawal, "Air time fairness for ieee 802.11 multi rate networks," in *IEEE Trans. Mobile Computing*, April 2008.