A Review of Software-Defined WLANs: Architectures and Central Control Mechanisms

Behnam Dezfouli, *Member, IEEE*, Vahid Esmaeelzadeh, *Student Member, IEEE*,
Jaykumar Sheth, *Student Member, IEEE*, and Marjan Radi, *Member, IEEE*

***Abstract*—The significant growth in the number of WiFi-enabled devices as well as the increase in the traffic conveyed through wireless local area networks (WLANs) necessitate the adoption of new network control mechanisms. Specifically, dense deployment of access points, client mobility, and emerging QoS demands bring about challenges that cannot be effectively addressed by distributed mechanisms. Recent studies show that software-defined WLANs (SDWLANs) simplify network control, improve QoS provisioning, and lower the deployment cost of new network control mechanisms. In this paper, we present an overview of SDWLAN architectures and provide a qualitative comparison in terms of features such as programmability and virtualization. In addition, we classify and investigate the two important classes of centralized network control mechanisms: (i) association control (AsC) and (ii) channel assignment (ChA). We study the basic ideas employed by these mechanisms, and in particular, we focus on the metrics utilized and the problem formulation techniques proposed. We present a comparison of these mechanisms and identify open research problems.**

***Index Terms*—Software-Defined Networking, IEEE 802.11, Architecture, Mobility, Interference, Centralized Algorithms**

I. Introduction

Wireless local area networks (WLANs) have been deployed broadly in various types of environments such as enterprises, universities, airports, shopping malls, and homes. The IEEE 802.11 standard, commonly known as WiFi, is becoming more popular and ubiquitous. In particular, the traffic demand of WiFi networks is increasing due to the emergence of high-definition video streaming, Internet of Things (IoT), cellular data offloading, online gaming, and virtual reality [1]–[6].

Statistics and forecasts show a continuous growth in the number of WiFi-enabled devices shipped as well as the amount of traffic conveyed through WLANs. Figure 1(a) shows the number of WiFi-enabled devices, including access points (APs), network interface cards (NICs), routers, switches, laptops, tablets, smartphones, and TVs that were shipped from 2012 to 2017. The shipment of WiFi-enabled devices was 1.58 billion units in 2012 and is estimated to be 4.91 billion units in 2017 [5]–[7]. Furthermore, the percentage of traffic conveyed by WiFi networks is growing continuously, as illustrated in Figure 1(b). During 2014, 41$\%$ of mobile data traffic was exchanged through WiFi networks, including the traffic offloaded from cellular networks into WLANs [4], [8], [9]. This traffic is expected to reach 53$\%$ by 2019. In addition, the scientific community has paid great attention to 802.11 networks, as demonstrated by the 165K citations that this topic has received until 2014, according to [10].

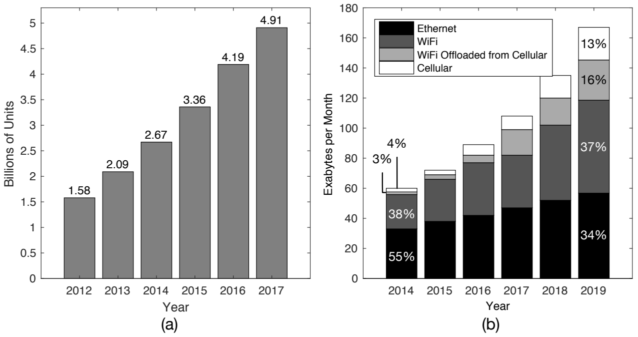

Fig. 1. (a) Increase in the number of WiFi-enabled devices shipped during 2012 to 2017. (b) Traffic growth of WiFi, cellular and Ethernet networks during 2014 to 2019 [8].

The increase in the number of WiFi-enabled devices and their growing traffic demand requires a dense installation of APs to improve spatial reuse and enhance network capacity. However, AP densification intensifies channel contention, increases the interference level among APs and their associated clients, and exacerbates the challenges of client handoff among APs. Therefore, effective association and interference control are both necessary to satisfy the QoS demands of clients in terms of throughput, delay, reliability and energy efficiency [11]–[15]. Satisfying these requirements is even more important and challenging when WLANs are used for mission-critical application such as industrial process control, factory automation, and medical monitoring. In particular, these applications require bounded packet delivery delay and high link reliability, and in some cases energy efficiency is a critical performance metric as well [16], [17].

In order to tackle the challenges of controlling complex wired networks (such as ISPs and data centers), *software-defined networking* (SDN) has been proposed to simplify and improve the design and development of network control mechanisms by decoupling the control and data plane. Instead of running network control mechanisms distributively on switching devices, a controller configures the switches to operate according to the decisions made centrally [18], [19]. Consequently, in addition to simplifying the deployment of

Behnam Dezfouli, Vahid Esmaeelzadeh and Jaykumar Sheth are with the Internet of Things Research Lab, Department of Computer Engineering, Santa Clara University, Santa Clara, CA, 95053 USA. E-mail: {bdezfouli, vesmaeelzadeh, jsheth}@scu.edu.

Marjan Radi is with Western Digital Corporation, Milpitas, CA, 95035 USA. E-mail: marjan.radi@wdc.edu.

network control mechanisms as applications running on the controller, these mechanisms can benefit from the controller's global network view. Not only for wired networks, SDN has gained popularity for the design of WLANs and cellular networks as well [20]–[23].

A *software-defined WLAN* (SDWLAN) enables central monitoring and control of network operation. A SDWLAN can be studied from two point of views, as follows:

- **Architecture**. A SDWLAN architecture: (i) defines the various network components (e.g., controller, APs, switches, middleboxes) used to build the network as well as the topology employed to connect these components; (ii) implements application programming interfaces (APIs) through which the controller communicates with network devices to perform data collection and distribute control commands; (iii) specifies the separation level between the control plane and data plane. The architectural features highly affect the development flexibility and performance of network control mechanisms [12], [24]–[26]. For example, when medium access control (MAC) functionalities are transferred from the APs to the controller, network control applications may implement AsC mechanisms with various capabilities and performance levels, depending on the separation level and APIs provided.
- **Control Mechanisms.** To benefit from the features of SDWLAN architectures, and in particular, the global network view provided, various control mechanisms, such as *association control* (AsC), *channel assignment* (ChA), transmission power control, and CCA threshold adjustment have been proposed by the research community. In this paper, we particularly focus on centralized AsC and ChA mechanisms because they have been widely proposed and adopted to benefit from the features of SDWLANs. AsC mechanisms address client-AP association for both static and dynamic clients to achieve various objectives, such as seamless mobility, fairness among clients, load balancing among APs, and interference mitigation. ChA mechanisms address the problem of channel assignment to APs in order to minimize the interference level experienced by APs and clients [27]–[29].

A. Contributions

At a high level, this paper studies SDWLANs from two perspectives: (i) SDWLAN architectures, and (ii) the two main central network control mechanisms, AsC and ChA, that are employed to control network operation. More precisely, the contributions of this paper are as follows:

- We first present an overview of software-defined networking and the standard protocols used for network monitoring, management and programming. Next, we review SDWLAN architectures and reveal their main objective, components, interconnection protocols, and APIs provided. Based on their main contribution, we classify architectures into the following categories: *observability and configurability*, *programmability*, *virtualization*, *scalability*, and *traffic shaping*. Then, we compare these architectures and highlight future research directions considering the current and upcoming requirements of SDWLANs.
- We review the AsC mechanisms proposed for SDWLANs. We first classify these mechanisms into two groups: (i) *seamless handoff*, mechanisms for reducing the overhead and delay of client handoff, and (ii) *client steering*, mechanisms to optimize parameters such as the airtime allocated to clients. Client steering approaches, in turn, are classified into two groups: (i) *centrally-generated hints*, where the controller relies on the global network view to generate hints for the association of clients, and (ii) *centrally-made decisions*, where the controller makes the association decisions and enforces the clients to apply them. In addition to discussing the impact of architecture on supporting seamless handoff, we compare the reviewed mechanisms in terms of optimization scope, traffic awareness, and the ability to recognize the QoS requirements of clients. By discussing the capabilities and weaknesses of AsC mechanisms, we present research directions towards designing AsC solutions for heterogeneous and dynamic SDWLANs with diverse QoS requirements. We also identify how the potential features of SDWLAN architectures can be employed to design more efficient AsC mechanisms.
- We review the ChA mechanisms proposed for SDWLANs. We classify these mechanisms into two groups, *traffic-aware* and *traffic-agnostic*, based on whether they incorporate APs' and clients' traffic into the decision making process. We specifically focus on how ChA mechanisms define and measure their interference metrics as well as the approaches employed for modeling and solving the formulated problems. Furthermore, we compare these mechanisms in terms of factors like client-awareness, dynamicity, and the unique features of different 802.11 standards. We propose research directions towards designing mechanisms that address the demands of emerging applications. We also identify how the potentials of SDWLANs can be employed to design more efficient ChA mechanisms.

When studying AsC and ChA mechanisms, we reformulate the problems using a consistent set of notations to ease the understanding and comparison of these mechanisms. Table I presents the key acronyms and notations used in this paper. Figure 2 shows the high-level organization of the paper.

II. Motivation

In this section, we overview the major application domains and requirements of WLANs, the challenges of wireless communication, and the importance of centralization in addressing the requirements of current and emerging applications.

A. Applications and Requirements

WLANs are being used in a variety of applications with various performance requirements. WLANs installed in enterprises, campuses and public areas usually serve a small to medium number of stationary and mobile users. For voice

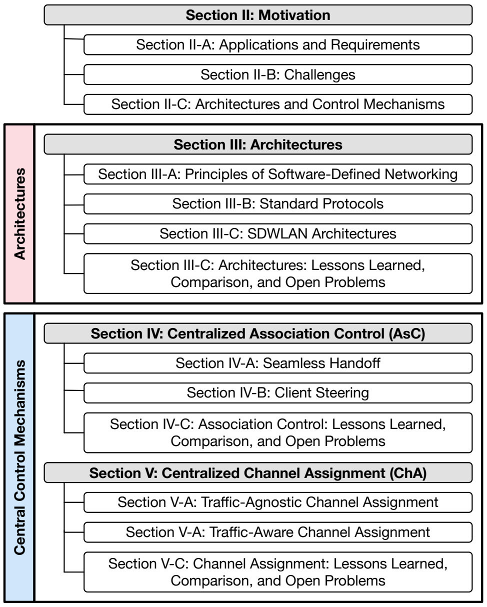
Section II: Motivation
Section II-A: Applications and Requirements
Section II-B: Challenges
Section II-C: Architectures and Control Mechanisms
Architectures
Section III: Architectures
Section III-A: Principles of Software-Defined Networking
Section III-B: Standard Protocols
Section III-C: SDWLAN Architectures
Section III-C: Architectures: Lessons Learned, Comparison, and Open Problems
Central Control Mechanisms
Section IV: Centralized Association Control (AsC)
Section IV-A: Seamless Handoff
Section IV-B: Client Steering
Section IV-C: Association Control: Lessons Learned, Comparison, and Open Problems
Section V: Centralized Channel Assignment (ChA)
Section V-A: Traffic-Agnostic Channel Assignment
Section V-A: Traffic-Aware Channel Assignment
Section V-C: Channel Assignment: Lessons Learned, Comparison, and Open Problems

Fig. 2. High-level organization of the paper.

TABLE I
KEY ACRONYMS AND NOTATIONS

Acronym	Definition
AsC	Association Control
AP	Access Point
API	Application Programming Interface
BSSID	Basic Service Set ID (AP's MAC address)
ChA	Channel Assignment
CCA	Clear Channel Assessment
DCF	Distributed Coordination Function
ISP	Internet Service Provider
LVAP	Light Virtual Access Point
MAC	Medium Access Control
NFV	Network Function Virtualization
NIC	Network Interface Card
PHY	Physical
RSSI	Received Signal Strength Index
SDN	Software-Defined Network
SDWLAN	Software-Defined WLAN
SINR	Signal to Interference-and-Noise Ratio
VAP	Virtual Access Point
WLAN	Wireless Local Area Network
Notation	**Definition**
$\mathcal{C} = \{c_1, ..., c_i\}$	Set of clients
$\mathcal{AP} = \{AP_1, ..., AP_i\}$	Set of access points
$\mathcal{CH} = \{ch_1, ..., ch_i\}$	Set of channels

and video streams, in addition to high throughput, providing an uninterrupted service for mobile users is an important performance metric [30]. For home WLANs, achieving these service requirements is more challenging due to the adjacency of multiple networks controlled individually [31], [32].

Industrial WLANs usually serve a larger number of nodes deployed for monitoring and control purposes. In addition, while most of the nodes are static, mobility is characterized by the speed and movement scale of robot arms, rotating parts, and mobile robots [33]. In these applications, the network infrastructure must satisfy a set of unique requirements [16], [34], [35], as follows: First, many of these applications require deterministic packet delivery delay. This is particularly important when sensors and actuators communicate with controllers because the delay of the control loop directly affects control performance. Second, a packet delivery reliability of 99% or higher must be guaranteed to ensure the robust operation of control and monitoring systems. In addition to sensors and actuators, industrial wireless networks also include high-rate devices such as cameras [36]. For example, it is important to ensure that the longer transmission duration and higher priority of video packets do not affect the timeliness of short control packets. Similar to industrial applications, other mission-critical systems, such as medical monitoring, must satisfy a similar set of requirements [17].

Besides user devices such as smartphones, more energy-constraint devices like sensors and actuators are being used nowadays in various types of applications [1], [17], [37]–[39]. For example, wireless sensors and actuators used in industrial environments, specifically those installed in high temperature and hard-to-access areas, are usually battery-powered. Furthermore, a greater number of low-power IoT devices and wearables are being used for applications such as medical monitoring. Therefore, energy is an important requirement of current and emerging applications of WLANs.

B. Challenges

Wireless networks introduce a new set of network control challenges, compared to wired networks. The broadcast nature of the wireless medium causes interference, resulting in link quality and throughput variations. For example, an AP may under-utilize its spectrum and hardware capacity due to the high interference level caused by neighboring APs or clients [40]–[42]. Interference level depends on *dynamic* factors such as mobility, traffic rate, number of devices, and environmental properties (e.g., path loss, multipath effect, obstacles, etc.) [11], [43]–[45]. The high path loss of wireless signals, and the inability of wireless transceivers to detect collisions, bring about new challenges such as hidden and exposed terminal problems [46].

From the network control point of view, *channel assignment* is a widely used mechanism to cope with interference, enhance throughput and reduce channel access delay. Channel assignment refers to the process of assigning RF channels to APs based on various factors, such as the interference relationship between APs and clients and the traffic demand of clients. The dynamics of wireless channel, the variability of user traffic, and variations in transmission power make channel assignment a challenging process. This process is further complicated by the introduction of channel bonding supported by 802.11n and 802.11ac standards. Channel bonding allows a device to combine multiple channels in order to transmit at a higher rate over a wider channel. However, the performance benefit of this technique depends on several factors and requires collaboration among APs [47], [48]. For example, using a larger bandwidth might increase contention with nearby APs and result in a longer channel access delay.

In addition to affecting interference level and the load of APs, *mobility* results in connection interruption because of the delay of re-association process [49], [50]. Although most of the large WLAN deployments employ a centralized user authentication scheme, the delay of association process is not completely eliminated. In particular, when a client associates with a new AP, the AP needs to establish the data structures pertaining to the new client. However, this process might result in communication interruption.

To increase the throughput of WLANs, extend coverage, and facilitate reliable handoff, *network densification* is employed in various types of environments such as enterprises and campuses. With this approach, the coverage range of the APs is reduced and the density of APs deployed per square meter is increased [14]. Despite the potential benefits of this approach, the network dynamics caused by interference and mobility are exacerbated by densification. For example, densification increases the number of re-associations as a client moves. In addition, due to the limited number of channels available in the 2.4 and 5GHz bands, channel assignment becomes a more challenging process.

From the energy point of view, the 802.11 standards provide two mechanisms: power saving mode (PSM), and automatic power saving delivery (APSD). However, the efficiency of these algorithms reduces as network density increases [51], [52]. For example, the concurrent wake-up of multiple clients results in collision and higher energy consumption [53]. Furthermore, energy is wasted when a node wakes up during a bad channel condition.

C. Architectures and Control Mechanisms

Central network control plays an important role to satisfy the requirements of existing and emerging applications. First, *architectures* are required to provide an infrastructure for centralized network control. Second, *centralized control mechanisms* are necessary to make control decisions based on network-wide dynamics and requirements. For example, utilizing centralized network control facilitates addressing the following challenges: To offer seamless handoff, we can eliminate the need for re-association by maintaining the association information of clients at a central controller; to enhance throughput, the controller can steer the association point of clients to balance the load of APs; the performance of channel assignment can be improved by utilizing centralization because channel assignment is, in essence, a graph coloring problem. In addition to client steering, seamless handoff, and channel allocation, which significantly affect throughput, delay, and energy consumption of clients, centralization enhances the adoption and performance of other techniques, such as scheduling, to further improve the aforementioned metrics [17], [52]–[54].

III. Architectures

In this section we overview the existing SDWLAN architectures. Before presenting these architectures, we first overview the basic concepts of software-defined networking and the standard protocols used for north and south-bound communication.

A. Principles of Software-Defined Networking

A software-defined network (SDN) decouples control mechanisms from data forwarding paths. In fact, a SDN has two planes: a (i) *data plane*, such as switches and APs, and a (ii) *control plane*, which is a controller that runs an operating system (e.g., NOX [55], Floodlight [56]) and mechanisms that control the network operation (e.g., AsC, ChA). The controller[1] communicates with data plane equipment through a south-bound protocol (e.g., OpenFlow [59], SNMP [60]), and exposes a set of north-bound APIs (e.g., REST) through which network control mechanisms are developed. In fact, the global network view of the controller and the reprogrammability of data plane significantly simplifies the design and deployment of network control mechanisms. Network control mechanisms are also referred to as *network applications*, as they are actually applications running on the network operating system.

SDN is an enabler of *network virtualization*, a.k.a., *network slicing*, which refers to the abstraction, slicing and sharing of network resources [61]. For example, an abstraction layer (e.g., FlowVisor [62]) is used to provide applications with an isolated view of resources. Network virtualization provides control logic isolation, as each application can see and control only the slice presented to it. Therefore, when a SDWLAN is shared by multiple operators, network slicing enables the enforcement of multiple access policies to the users associated with various network operators. In addition, virtual networks corresponding to various services can be established to support differentiated services and achieve higher QoS control over resources. Virtualization also expedites the design and development of novel wireless technologies. For example, one or multiple experiments can be run on slices of a wireless network while the production network is in operation. Three levels of slicing are applicable to a SDWLAN: (i) Spectrum (a.k.a., link virtualization): requires frequency, time or space multiplexing. (ii) Infrastructure: the slicing of physical devices such as APs and switches. (iii) Network: the slicing of the network infrastructure.

Given the simplified configuration of data forwarding paths, SDN promotes the use of *network function virtualization* (NFV). Instead of using dedicated hardware, NFV relies on the implementation of services using general computing platforms. NFV is the virtualization of network functions such as domain name service (DNS), firewall, intrusion detection, load balancing and network address translation (NAT). To this end, a network service is broken into a set of functions that are executed on general purpose hardware. These functions can then be introduced to or moved between network devices whenever and wherever required. As we will show later, software-defined radio (SDR) and virtual APs (VAPs) are the two most notable types of NFV in SDWLAN to offload the processing of APs into general computing platforms.

B. Standard Protocols

In SDN architectures, the controller communicates with the data plane through a south-bound protocol, and provides north-bound APIs for application developers. To avoid exposing the complexities of south-bound APIs, the north-bound APIs are usually REST-based [63], [64], and network engineers use

[1] To achieve reliability, a logical controller may represent multiple physical controllers [57], [58].

language abstractions, such as Frenetic [65], Pyretic [66], and NetCore [67], for application development [68]. For example, FloodLight [56] and OpenDaylight [69] both support REST north-bound APIs. In the rest of this section we review the widely-adopted standard south-bound protocols.

Simple Network Management Protocol (SNMP) [60], [70]–[72]. This is an application layer protocol used to monitor and configure network elements. SNMP exposes *management data* in the form of variables on the *managed systems*. The nature of the communication is fetch-store: either the manager fetches resources or stores the value of an object on the agent. SNMP cannot directly implement actions. For example, to restart an agent, the manager cannot send a command to reboot the agent; instead, it sets the value of the reboot counter, which indicates the seconds until next reboot. SNMP has been the most widely used management protocol in production networks due to its simplistic nature and ease of usage. Although SNMPv3 introduces significant security enhancements, SNMP has been mostly used for monitoring, rather than configuration [73]. Furthermore, due to the sequential execution of commands, SNMP introduces high network overhead and may result in network failure in large-scale deployments [74], [75].

Netconf [76]. Netconf uses remote procedure call (RPC) to exchange messages and apply configuration through Create, Retrieve, Update, and Delete (CRUD) operations. In addition, it enables the manager to define what action (e.g., continue, stop, roll-back) should be taken if a command fails. Netconf has been mostly used for configuration, and the assumption is that another protocol such as SNMP is used for monitoring purposes [73]. YANG is a tree-structured modelling language used by Netconf to describe the management information of network elements. Since YANG enables the definition of data models, Netconf has been widely used by industry; however, vendors may not support a consistent set of data models.

OpenFlow [59]. This protocol assumes that data plane elements are simple packet forwarding devices and their operation is determined by the forwarding rules received from the controller. Whenever a new flow enters a switch, the forwarding table of the switch is queried to find a matching forwarding rule. If no forwarding rule is found, either the packet is dropped or an inquiry is sent to the controller to retrieve the required *action*. OpenFlow 1.1 replaces actions with *instructions*, which enables packet modification and updates actions. The possibility of connecting to multiple controllers has been introduced in OpenFlow 1.2. OpenFlow 1.3 enables per-flow rate control.

Control And Provisioning of Wireless Access Points (CAPWAP) [77]. This protocol enables a controller to establish DTLS [78] connections with APs to exchange data and control messages. Data messages encapsulate wireless frames, and control messages are used for monitoring and control purposes. In addition to centralizing authentication and network management, CAPWAP defines two MAC operation modes [79]: *local MAC* (LM) and *split-MAC* (SM). Using the LM mode, most of 802.11 MAC operations are implemented in APs and the role of controller is almost negligible. Using the SM mode, the real-time operations of 802.11 MAC are run on APs, and the rest of operations, such as the generation of beacons and probe responses, are handled by the controller.

CPE WAN Management Protocol (CWMP) [80] (a.k.a., TR-069). Published by Broadband Forum, CWMP is a text-based protocol for communication between Auto Configuration Servers (ACS) and Customer Premise Equipment (CPE) such as modems, APs and VoIP phones. The primary features of CWMP are secure auto-configuration, dynamic service provisioning, software/firmware image management, status and performance monitoring, and diagnostics. One of the main capabilities of CWMP is to enable the devices behind NAT to securely discover ACSs and establish connection. In addition, an ACS can request a session start from a CPE. The commands (e.g., get, set, download, upload, etc.) run over a SOAP/HTTPS application layer protocol. In fact, a CPE acts as a client and the ACS is the HTTP server. Configuration and monitoring of CPEs is performed by setting and getting the device parameters, which are defined as a hierarchical structure. CWMP is considered to be an important management protocol in M2M architectures [81], [82]. However, this protocol does not address client mobility in WLANs.

As OpenFlow has been primarily designed to configure the flow table of switches, an additional protocol, such as SNMP, Netconf, or a proprietary protocol, should be used to enable the control of wireless devices, as we will see in Section III-C. Therefore, both SNMP and Netconf are widely supported by SDN controllers (e.g., OpenDaylight, ONOS [83]). We provide further discussion about protocols in Section III-D1.

There are other protocols, in addition to those mentioned in this section, used for south-bound communication. For example, although REST APIs are mostly used for north-bound interactions, REST is also employed for south-bound communication by controllers such as Ryu [84].

C. SDWLAN Architectures

In this section we review the architectures proposed for SDWLANs. We categorize these architectures based on the main contributed feature into five categories: *observability and configurability*, *programmability*, *virtualization*, *scalability*, and *traffic shaping*. However, it should be noted that some of these architectures present multiple features, as we will show in Figure 7 and Table II. Please note that the AsC and ChA mechanisms proposed to benefit from these architectures will be studied in Section IV and V, respectively.

*1) **Observability and Configurability**:*

The architectures of this section provide the means for central monitoring and configuration, however, they cannot be used to develop new network control mechanisms.

DenseAP. This architecture [12] addresses the challenges of densely-deployed WLANs in terms of AsC and ChA. The two main components of this architecture are *DenseAP APs* and *DenseAP Controller*. DenseAP APs are off-the-shelf PCs running Windows operating system. Each AP runs a *DenseAP daemon*, which is a user-level service responsible for managing AP functionality. This daemon monitors NIC and reports to DenseAP Controller periodically. A report includes the list of associated clients, sample RSSI values, traffic pattern of clients, channel condition, and a list of new clients requesting to join the network. The DenseAP Controller enables the user to tune parameters, makes control decisions, and informs the daemon to apply the configuration. We will review the AsC mechanism of DenseAP in Section IV-B2.

Cisco Unified Wireless Network (CUWN). CUWN [85] proposes an architecture and a set of configurable services, including seamless mobility control, security and QoS provisioning. CUWN uses either Lightweight Access Point Protocol (LWAPP) [86] or CAPWAP [77] as its south-bound protocol. As supported by CAPWAP, CUWN provides two MAC modes: *local MAC* (LM) and *split-MAC* (SM). The Radio Resource Management (RRM) [87] of CUWN provides several network control services. For example, RRM establishes *RF groups* and determines a leader controller for each group. To this end, each AP periodically transmits Neighbor Discovery Protocol (NDP) messages on all channels. NDP messages contain information such as the number of APs and clients. All APs forward the received NDP messages to the controller, which establishes a network map, groups APs into RF Groups, and chooses a leader controller for each group. We will review the AsC and ChA mechanisms of CUWN in Section IV-A and V-B, respectively.

*2) **Programmability**:* In addition to central monitoring, these architectures expose north-bound APIs that enable the implementation of control mechanisms as applications running on a controller. We overview these architectures as follows.

DIRAC. This architecture [88] proposes a proprietary south-bound protocol using two main components: a *router core* (RC) running on a PC, and *router agents* (RA) running on APs.

RAs receive messages from RC and convey them to the NIC's device driver. There are three types of interactions between the RC and RAs: *events*, *statistics*, and *actions*. Sample events are association, re-association, and authentication, which are reported by RAs to the RC. RAs periodically report statistics, such as packet loss rate and SNR. The RC enforces its policy by sending actions to RAs. For example, the RC may use `set_retransmissions()` to limit the number of retransmissions. The control plane of the RC has two main components: (i) *management mechanisms*, which are the network applications, and (ii) a *control engine*, which includes components to simplify network application development.

Trantor. This architecture [89] is an improved version of DenseAP [12], and its south-bound protocol extends the exchange of control messages with clients (in addition to APs). These messages include packet loss estimation, the RSSI of packets received from APs, channel utilization, and neighborhood size. Trantor also supports "active monitoring", through which clients and APs are directed to exchange packets to measure metrics that are used by network control mechanisms. In addition, Trantor provides a set of APIs used to control the association, channel, transmission rate and transmission power of devices. For example, using `associate` (AP_i) a client is instructed to associate with AP_i. Unfortunately, no network control mechanism has been proposed to benefit from these primitives.

Dyson. This architecture [90] (which is an extension of Trantor) enables both clients and APs to communicate with the controller. Clients are categorized into two groups: *Dyson-aware clients* and *legacy clients*. Only Dyson-aware clients can communicate with the controller. The measurement APIs can be used to collect information such as: the sum of the RSSI values (total RSSI) of all received packets during a measurement window, the number of transmitted packets per PHY rate, the number of transmission failures, and the channel airtime utilization.

Dyson establishes a network map based on the information collected from APs and Dyson-aware clients. The network map provides the following components: (i) *node locations*: reflects the location of APs (which are fixed) and clients (using [91]); (ii) *connectivity information*: a directed graph that shows from which APs/clients an AP/client can receive packets; (iii) *airtime utilization*: reflects channel utilization in the vicinity of each node; (iv) *historical measurement*: a database to store measurements.

Dyson's APIs include commands to configure the channel, transmission power, rate and CCA of APs and clients. The API set also includes commands to manage AP-client associations. A reduced set of control APIs is used to support legacy clients. This set enables the controller to control the association point and operating channel of these clients.

ÆtherFlow. This architecture [92] simplifies data plane programmability by extending OpenFlow based on its formal specifications. The control interfaces are categorized according to the OpenFlow specification as follows: (i) *Capabilities*: interfaces through which a controller inquires the capabilities of an AP's radio interfaces. These capabilities include the number of channels, transmission power levels, etc. (ii) *Configuration*: these interfaces enable the configuration of physical ports (e.g., channel, transmission power) and logical wireless ports (e.g., SSID, BSSID). (iii) *Events*: interfaces for event reporting (e.g., probe, authentication, association) to a controller. (iv) *Statistics*: these interfaces enable a controller to query wireless related statistics. ÆtherFlow has been implemented in CPqD SoftSwitch [93], which is installed on OpenWRT [94]. The performance and capability of this architecture has been evaluated using a predictive AsC mechanism, which we will explain in Section IV.

COAP. Coordination framework for Open APs (COAP) [31] is a cloud-based architecture for residential deployments. In addition to using OpenFlow, this architecture proposes an open API to provide a cloud-based central control of home APs. COAP improves client QoS by running central control algorithms as well as enabling cooperation among APs. A COAP AP implements three modules: (i) *APConfigManager*, (ii) *DiagnosticStatsReporter*, and (iii) *BasicStatsReporter*. APConfigManager is used for AP configuration, and the other two modules provide diagnostic and basic statistics for the cloud-based COAP controller. The COAP controller has three modules: (i) *StatsManager*, (ii) *COAPManager*, and (iii) *ConfigManager*, which are implemented within Floodlight. The controller collects the required information using the StatsManager module and performs control configurations through ConfigManager. The COAPManager module allows network administrators to implement new strategies and policies.

ResFi. [32] enables information collection and sharing in

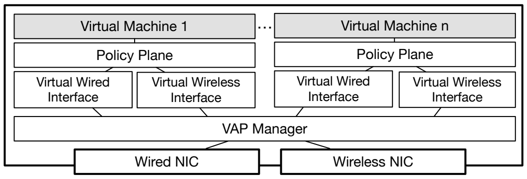

Fig. 3. The architecture of an AP used by AP Aggregate [101].

residential networks. *ResFi Agent* is a user-space program that runs on APs and provides IP connectivity over the wired backhaul network. This agent instructs the APs to exchange messages through which APs discover each other's IP address. After the discovery phase, each AP establishes a secure point-to-point connection to the agents of discovered APs using the wired backbone. Each agent can then share information with other APs or a remote controller. For example, an AP can subscribe to the publish socket of its neighbors to receive their updates. Using ResFi Agents on an AP does not require any kernel or driver modification as it simply relies on the interfaces provided by hostapd [95]. The APIs provided enables both distributed and centralized management of residential WLANs.

*3) **Virtualization**:* In this section we review the architectures that offer virtualization to support seamless mobility, slicing, and NFV. In particular, we discuss *AP virtualization*, which refers to the mapping of multiple virtual APs to a physical AP (e.g., Odin [49], CloudMAC [50]), or mapping a virtual AP to multiple physical APs (e.g., SplitAP [96], BIGAP [97]).

WVT. Wireless Virtualization Testbed (WVT) [98] relies on AP virtualization to run isolated experiments on a testbed. Corresponding to each experiment, each physical AP runs a *Node Agent*, where each Node Agent is controlled by a *Node Handler* running on the controller. Node Handlers provide the user with APIs for monitoring and controlling Node Agents. In addition to Node Agents, the APs run a *Node Overseer*, which is responsible for activating the Node Agents in a round robin fashion using TDMA scheduling. In fact, at each slot boundary, the Node Overseer records the state (e.g., SSID, channel) of the currently running Node Agent and restores the state of the next Node Agent. Node Overseers can be configured through the *Master Overseer* running on the controller. This architecture uses an NTP time synchronization server, which is required for TDMA scheduling.

OpenRoads. This architecture [99], [100] adds OpenFlow to APs. In addition, due to the limitations of OpenFlow, OpenRoads uses SNMP for configuring the APs' parameters. Resource slicing is provided through using FlowVisor and SNMPVisor; the former slices data path, and the latter slices configuration commands through sending them to the appropriate data path element.

AP Aggregate. [101] presents a flexible architecture that enables the aggregation of multiple virtual APs (VAPs) inside an AP, as well as slicing resources on APs. VAP aggregation is also used for interference reduction by turning off lightly-loaded APs. Figure 3 shows the architecture of a physical AP. The *VAP Manager* is responsible for creating and destroying VAPs based on the commands received from the controller. The *Policy Plane* is a set of settings that are applicable to the VAP. For example, these policies may specify the QoS and firewall settings of the VAP. A *Virtual Wireless Interface* has a SSID, MAC and IP address. Multiple Virtual Wireless Interfaces are multiplexed into the Wireless NIC using TDMA. Based on its collected information, a controller moves VAPs between APs. This architecture employs layer-2 tunneling techniques when VAPs are moved between subnets.

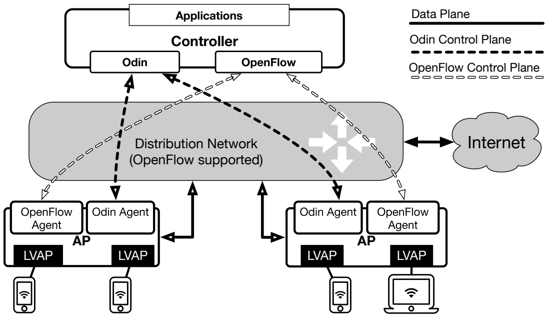

Fig. 4. Odin [102] introduces the concept of lightweight virtual APs (LVAP) to reduce the overhead of handoffs.

Odin. Figure 4 shows the architecture of Odin [49], [102]–[104]. This architecture is composed of the following components: (i) *Odin Controller*: maintains a global network view including the status of APs, clients and OpenFlow switches. (ii) *Odin Agents*: run on APs, and enable communication through a proprietary south-bound protocol. Time-critical operations (e.g., ACK transmission) are performed by APs, and non-time-critical operations (e.g., association) are handled by the controller. This operation results in a split-MAC protocol, which is inspired by CAPWAP. (iii) *Applications*: implemented on the controller. Applications may use the information provided by Odin Agents, OpenFlow, and SNMP. The APIs support the implementation of various control mechanisms such as AsC, load balancing and hidden-terminal handling.

Odin introduces the concept of *light virtual AP* (LVAP) to separate clients' association states from physical APs; thereby reducing the overhead of handoffs. An LVAP is characterized by the following four-tuple:

- client IP address,
- a unique virtual basic service set ID (BSSID),
- one or more service set ID (SSID),
- a set of OpenFlow rules.

For each client, an LVAP resides in an AP (hosted by Odin Agent) to represent client-AP association. When the controller decides to associate a client with a new AP, it can simply transfer the LVAP of that client to the new AP. From a programmer's point of view, multiple clients are connected to different ports (the LVAPs) of a physical AP. LVAPs also simplify client authentication through adding a session key to the client's LVAP. We will discuss about Odin's AsC mechanism in Section IV-A.

SplitAP. [96] proposes an architecture to support network virtualization and manage clients' share of airtime, especially for uplink traffic. Each AP runs a *SplitAP controller*, which

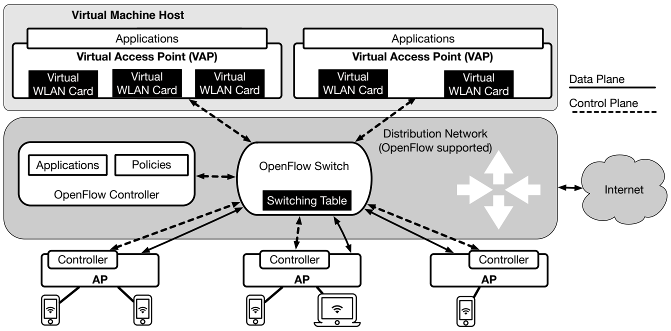

Fig. 5. CloudMAC [26] reduces the complexity of APs through running VAPs on a cloud computing infrastructure.

is responsible for VAP management and computing the uplink traffic of each slice. By relying on the VAP concept, each physical AP broadcasts the beacons of independent virtual networks. For example, when three different ISPs utilize an infrastructure, each AP emulates three VAPs. This architecture requires client modification to enforce controller commands. Specifically, each client runs a *client controller* that adjusts traffic based on the commands received from the associated AP. When a SplitAP controller realizes that the usage of a particular slice is higher than its threshold, the AP broadcasts a new maximum uplink airtime utilization that can be consumed by clients in that slice. All the APs are connected to a shared backhaul, through which they receive channel allocation commands from a controller.

VAN. Virtualized Access Network (VAN) [105] proposes an architecture for central control of residential networks. The architecture is composed of the following components: (i) *residential APs*, (ii) a *controller*, which is part of the ISP, and (iii) *content provider servers* that are part of the content provider network (e.g., Netflix). The controller provides a set of APIs through which content providers control home networks. These APIs are suitable for different traffic classes such as video streaming and file transfer. For example, when a content provider receives the request of a user, the content provider communicates with the controller and reserves resources for the requested data flow.

In this architecture, IP address assignment and authentication are performed centrally per user device; therefore, it is possible to transfer the association of a user device between APs. More importantly, this architecture enables the sharing of APs' bandwidth between neighbors. Given a flow bandwidth requirement and the neighborhood of each client, the controller associates clients to APs by employing a heuristic algorithm.

CloudMAC. Figure 5 illustrates the CloudMAC [26], [50] architecture. CloudMAC decomposes AP operations into two modules and employs OpenFlow switching tables to manage the communication between these modules. The main components of CloudMAC are (i) APs, (ii) VAPs, (iii) an OpenFlow switch, and (iv) an OpenFlow controller. APs only forward MAC frames. All other MAC functionalities are performed by VAPs that reside in a cloud computing infrastructure. VAPs are operating system instances on a hypervisor such as VSphere. Each VAP can include multiple virtual WLAN cards, where each card is implemented as a driver in a VAP; therefore, standard software tools can utilize these cards. Virtual machines are connected to physical APs by an OpenFlow-based distribution network. The generation and processing of MAC frames are performed by VAPs. A physical AP, which is a slim AP, sends and receives raw MAC frames from clients and handles time-sensitive operations (e.g., ACK generation and re-transmission).

APs and VAPs are connected by layer-2 tunnels [106] and the OpenFlow switch. The forwarding table of switches are used to decide how packets must be routed between physical APs and VAPs. This mechanism simplifies the implementation of control mechanisms such as AsC. For example, while Odin requires LVAP migration to handle mobility, CloudMAC simply reconfigures the switching tables to change the data forwarding path.

When a control command is generated by a network application, the command is processed by VAP and a configuration packet is forwarded toward the OpenFlow switch. The OpenFlow switch forwards the packet to the OpenFlow controller to check the legitimacy of the configuration command according to user-defined policies. If the command is legitimate, the OpenFlow controller forwards the packet to the AP.

To study the performance of CloudMAC, the round trip time (RTT) is measured between a client and a VAP. Due to the processing overhead of the OpenFlow switch and the delay overhead added by the tunnels, the average RTT is increased from 1.79 ms to 2.28 ms, compared to a standard WLAN. However, time-critical MAC frames (e.g., association response frames) are delivered fast enough to satisfy the timeliness requirements. For large TCP packets, throughput is decreased by almost 8.5%, compared to a standard WLAN. This performance degradation is because of the tunnels implemented in user space, which requires context switching.

EmPOWER. EmPOWER [107] relies on the concept of LVAP (proposed by Odin) in order to decrease the overhead of client mobility. The controller runs Floodlight as the operating system and FlowVisor to enable virtualization. In this architecture, each AP is equipped with an Energino [108] add-on, which is an open toolkit for energy monitoring. This add-on provides REST-based APIs that enable network administrators to turn on and off the APs.

EmPOWER2. An improved version of EmPOWER has been proposed in [24], [109]. We refer to this architecture as EmPOWER2. This architecture provides a full and open set of Python-based APIs by introducing four key abstractions: (i) *LVAPs*: a per-client VAP similar to Odin. (ii) *Resource pool*: each resource block is identified as $((frequency, bandwidth), time)$. For example, an AP working on channel 36 with bandwidth 40MHz is represented as $((36, 40), \infty)$. Similar notation is used to represent the 802.11 standard supported by LVAPs. This representation provides a mean for LVAP to AP mapping. (iii) *Channel quality and interference map*: provides network programmers with a global network view in terms of the channel quality between APs and LVAPs. This view enables the programmers to allocate resources in an efficient manner. (v) *Port*: specifies the configuration of an AP-client link in terms of power, modulation and MIMO configuration. We will study its AsC and ChA mechanisms in Section IV-B2 and V-A, respectively.

Sd-wlan. The main idea of Sd-wlan [110], [111] is the implementation of most MAC functionalities at a controller, which is essentially similar to Odin [102]. To this end, an extended version of OpenFlow is proposed as the south-bound protocol. Using this protocol, the controller can instruct the APs whether they should send ACK packets or not. When a client needs to change its point of association, the controller must install new rules on the new AP. The controller also updates OpenFlow switches to direct the traffic being exchanged between the controller and APs.

BeHop. The main components of BeHop [112] are: (i) *BeHop APs*, (ii) a *BeHop collector*, and (iii) a *BeHop controller*. BeHop collector is responsible for monitoring and collecting information from the network. The controller includes network control mechanisms, in addition to processing and responding to probe, authentication and association requests. The collector may be implemented internally (inside the controller) or externally as a separate component.

A proprietary south-bound protocol has been implemented by running *monitoring agents* on APs. This protocol is used to collect and forward statistics (e.g., channel utilization, SNR, RSSI, and PHY rate) to the collector. The controller communicates with the collector through a remote procedure call (RPC) interface to update its status about BeHop APs.

BeHop APs create a VAP per client, and each AP maintains a *client table*. Each entry of the client table includes client control information such as client-VAP mapping and client data rate. VAPs can be added to or removed from APs using the south-bound protocol implemented. Each BeHop AP operates as an OpenFlow switch and exposes APIs for controlling channels, power, and association. APs forward control traffic (e.g., probe, association) to the controller for further processing.

RCWLAN. This architecture [113] addresses the coexistence of multiple virtual networks (slices). For each virtual network, a VAP is created on the physical APs, where each VAP has its own virtual machine and MAC queue. The set of resources allocated to each slice is controlled by configuring the MAC parameters of the VAPs. The extension of RCWLAN is vBS, which we explain next.

vBS. [114] supports service-based wireless resource reservation, which is similar to the concept of flow-based virtualization [62]. Specifically, by reserving resources for time-critical services (e.g., VoIP), vBS provides QoS guarantees even in the presence of background traffic. In contrast with SplitAP, which supports virtualization through broadcasting different ESSIDs, this architecture relies on the concept of *virtual Base Station* (vBS). A vBS is a virtual multichannel AP that uses the resources of multiple physical APs. All of the APs share the same BSSID (MAC address) and ESSID; therefore, AP selection and handoff are completely handled by the controller. When a network administrator registers a new service, the controller creates a vBS for that service. When a client joins the network for the first time, the controller associates the client with an AP of the common vBS. The common vBS provides a service for non-prioritized traffic. When the client initiates a service, the controller performs a client handoff to the vBS of that service. The association status of the client is transferred to the new AP, and the OpenFlow switch is updated.

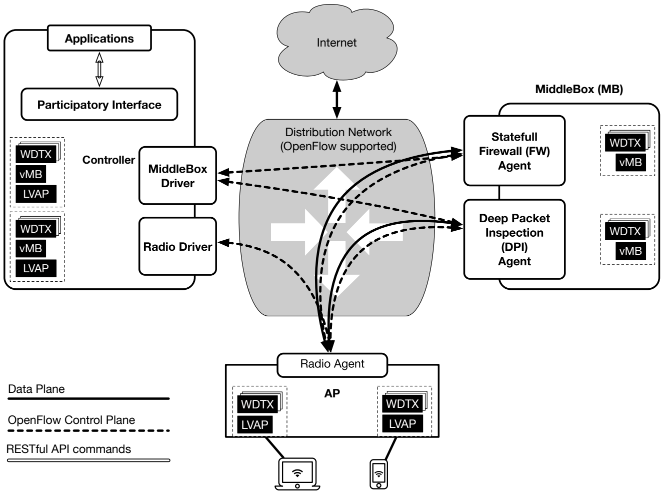

Fig. 6. OpenSDWN [25] enhances data plane programmability by introducing virtual middle boxes (vMBs).

BIGAP. The main goal of this architecture [97] is to support seamless mobility by mapping physical APs to a single VAP. BIGAP assumes that each AP has two wireless interfaces: one for serving as AP, and the second one for statistics collection which works in promiscuous mode to periodically overhear packets on all channels. The collected information is used by the controller to provide a fast AsC mechanism, which will be explained in Section IV-A. In order to reduce the complexities of the handoff process, BIGAP requires all of the APs to share the same BSSID. However, to avoid collisions, BIGAP requires each AP to select a channel that is not being used by its two-hop neighborhood. Association of a client with another AP is achieved through a channel switch command and transferring client's status to the new AP. BIGAP provides a rich south-bound protocol for statistics collection and association control. The AP-related APIs are implemented by modifying hostapd [95], and the APIs are available as RPC by relying on ZeroRPC [115] and ZeroMQ [116].

OpenSDWN. Figure 6 illustrates this architecture [25]. Similar to Odin, OpenSDWN provides per-client VAP; however, the new concepts of OpenSDWN are *virtual middle boxes* (vMB) and *wireless datapath transmission* (WDTX) rules. A vMB encapsulates a client's middle box state. Due to the small size of vMBs, their transfer across network does not introduce any significant overhead. Each vMB includes the following information: (i) the list of tunable parameters of the client, (ii) the states of client's active connections, (iii) the packet-based statistics, and (iv) an event list that defines the behavior of the client.

The controller implements a MB driver and Radio Driver to communicate with the agents of MBs and APs, respectively. Through the Radio Driver, the controller manages LVAPs and collects the status of clients. Furthermore, the controller can get/set per-flow transmission rules through its radio interface. To this end, OpenFlow rules are combined with WDTX rules within APs. WDTX rules provide per-flow service differentiation using a newly defined action that is compatible with OpenFlow protocol. The controller manages vMBs, gathers the statistics of vMBs, and receives the required events from different MBs. The latter, in particular, is supported through a publish/subscribe interface to ensure controller update when specific events occur.

A middlebox in OpenSDWN can have two types of interfaces: (i) a *stateful firewall* (FW) agent and (ii) a *deep packet inspection* (DPI) agent. The FW agent keeps track of all traffic passing through the MB in both directions and provides statistics per client and per flow. Also, the FW agent runs the WDTX rules defined by the controller for different flows and frames. The DPI agent is responsible for detecting events such as denial of service (DoS) attacks. The agents of a MB generate and send events that are of the interest to the controller. For instance, a significant change in the load of an AP may trigger sending a report to the controller. The controller can submit a report list to a MB in order to obtain only requested events.

The OpenSDWN controller exposes a participatory interface that enables network applications to define flow-based and client-based priorities. This is achieved by providing RESTful APIs through which LVAPs, WDTXs and vMBs are monitored and controlled.

Through empirical evaluations, the authors showed the small delay overhead of vMB migration among MBs to support mobility, and they demonstrated the effectiveness of using WDTX rules to perform service differentiation.

*4) **Scalability**:* Scalability depends on topology, as well as factors such as the overhead of south-bound protocol, the virtualization mechanisms employed, and the traffic and mobility pattern of clients. The scalability of SDWLANs is particularly important due to the dynamic nature of these networks. Specifically, an architecture with a high control plane delay cannot be used for quick reactions to network dynamics.

AeroFlux. This architecture [117], [118] highlights that per-flow or per-packet control mechanisms require short and bounded control plane delay. For example, implementing rate and power control mechanisms in a controller may result in an overloaded and high-latency control plane. To address this concern, AeroFlux employs a 2-tier control plane: *Global Control* (GC) plane, and *Near-Sighted Control* (NSC) plane. GC handles non real-time tasks and operations that require a global network knowledge. For example, authentication, large-scale mobility, and load balancing are handled by the GC. NSCs are located close to the APs to perform time-critical operations, such as rate control and power adjustment, per packet. For example, in the case of video streaming, AeroFlux can configure APs to use lower rates and higher transmission power values for key frames, compared to regular frames.

CUWN. This architecture [85] employs split-MAC as well as a multi-tier controller topology to address scalability. Based on the RF locations of APs, CUWN establishes *RF groups* and determines a leader controller for each RF group. All APs connected to a controller belong to an RF group. In this way, the controllers are aware of RF location of APs and their interference relationship.

Odin and CloudMAC. These architectures [49], [50] rely on the the split-MAC technique proposed by CAPWAP. In other words, to mitigate the negative effect of controller-AP communication, time-critical operations are handled by APs, and non time-critical operations are offloaded to the controller. In addition to utilizing split-MAC, during an association, CloudMAC prevents the overhead of moving VAPs between APs through configuring the switches of the distribution network to forward the client's data to another AP.

*5) **Traffic Shaping**:* The architectures of this section offer more than the regular programmability used for configuring APs and switches. In fact, these architectures extend data plane programmability by enabling traffic shaping, which is used for purposes such as scheduling and scalability.

CENTAUR. [119] can be added to SDWLAN architectures to improve the operation of the data plane in terms of channel access and contention resolution. When data traffic passes through a controller or programmable middleboxes, data plane programmability can be supported without AP modification. To this end, CENTAUR proposes central packet scheduling mechanisms to avoid hidden-terminal transmissions and exploit the exposed-terminal condition. For example, when the transmission of two APs to their associated clients may cause a hidden-terminal collision, the controller carefully adjusts the interval between packet transmissions to avoid the problem. A salient feature of CENTAUR is that it requires minor changes to APs and no client modification is required. CENTAUR improves UDP and TCP throughput by around 46% and 61.5%, compared to DCF [120].

CloudMAC. This architecture [26] is capable of implementing downlink scheduling by configuring OpenFlow switches to use simple rate shaping or time division algorithms. For instance, during a time slot, the switch may forward only the packets of a specific physical AP while queuing the packets of other APs. Switching rules may be changed per time slot to provide a time division protocol.

Ethanol. Ethanol [121] is similar to its predecessors (e.g., [26], [49]) in terms of implementing slim APs and shifting most of the MAC functionalities to a controller. However, Ethanol argues that the original OpenFlow protocol cannot be used for QoS provisioning in wireless networks. Accordingly, to complement OpenFlow, the Ethanol architecture proposes a customized protocol, named *Ethanol protocol*, to provide control interface for wireless components and QoS control. Therefore, an Ethanol AP provides two interfaces: (i) Ethanol, through an *Ethanol Agent,* and (ii) OpenFlow. The Ethanol Agent provides APIs for QoS control on the APs. Ethanol uses the APIs provided by the Ethanol Agent to exploit Hierarchical Token Bucket (HTB) [122] scheduling in order to perform per-flow programmability. Specifically, different queues are defined on Ethanol APs. When a flow arrives, it is assigned to the proper queue based on its class of service (e.g., voice, video).

Ethanol [121] provides the details of API implementation following an object-oriented approach. *Entities* are defined as physical or virtual objects that could be configured or queried. Each entity has its own properties. For example, an AP is a physical entity that includes properties such as beaconing interval. A flow is a virtual entity that includes properties such as a packet counter. The controller communicates with entities through their get/set interfaces. Entities may also include events, which trigger the controller to take proper actions. Ethanol API has been implemented on OpenWRT [94] by exploiting Pantou [122], a software package enabling OpenFlow on OpenWRT.

D. Architectures: Learned Lessons, Comparison, and Open Problems

Figure 7 highlights the main features and Table II summarizes the properties of reviewed architectures. In this section, we summarize the learned lessons, study the proposed features and identify future research directions and potential solutions.

1) ***Programmability****:* A *reconfigurable architecture* only enables the adjustment of parameters pertaining to a set of predefined network control mechanisms. For example, CUWN's APs are like regular APs with added CAPWAP support; thereby, it is not possible to implement a new AsC mechanism as CUWN employs a set of proprietary mechanisms for network control. In fact, we can argue that CUWN and DenseAP introduce a *management plane* instead of a control plane. On the other hand, a *programmable architecture*, such as OpenSDWN and Odin, provides north-bound APIs through which control mechanisms running on the network operating system are developed.

After the standardization of OpenFlow, most SDWLAN architectures have included this standard in their south-bound communication, as Table II shows. However, this protocol has been mainly designed to configure the flow tables of switches; therefore, it cannot be used for the configuration of wireless data plane equipment. Furthermore, since OpenFlow is very low-level, too many implementation details are exposed to network programmers [24]. Therefore, we can observe the introduction of proprietary protocols (e.g., Odin, Ethanol) and the extensions of OpenFlow (e.g., AeroFlux, ÆtherFlow). However, this results in interoperability issues. Another shortcoming of OpenFlow is the lack of supporting transactions, which means the operation of a network device during receiving updates is unpredictable. This has a particular implication on wireless network control mechanisms. For example, when an AsC mechanism sends updates to multiple APs, network performance may significantly drop during the update process due to the inconsistency of APs' software. Despite the significant number of studies on the performance and improvement of OpenFlow for wired networks [57], [123]–[125], there is a very limited study on the use of OpenFlow and other standard protocols (e.g., CWMP, SNMP, Netconf) for SDWLAN design [24], [92], [126], [127]. In addition, it is not clear what programmability features are required to develop architectures that include recent high throughput standards. For example, supporting the 802.11ad [128] standard requires additional interfaces for central coordination and training of antennas.

Our review shows that data plane programmability is supported at four levels: controller (e.g., CENTAUR), switches (e.g., CloudMAC), APs (e.g., Ethanol), and clients (e.g., Dyson). While controller-based approaches provide higher flexibility, they pose scalability and processing challenges. Specifically, the variations of data plane delay may inadvertently affect the accuracy of implemented mechanisms. For example, deterministic controller-AP delay is an essential requirement when packet scheduling is implemented at a controller in order to avoid collisions caused by concurrent downlink transmissions. On the other hand, AP-based approaches require low and bounded control plane delay to ensure prompt enforcement of the decisions made centrally. Meanwhile, the programmability of switching equipment can be employed to improve scalability and cope with overhead issues. For example, when a client is associated with a new AP, CloudMAC configures the switches of the distribution network to forward the client's data to another AP, thereby preventing the overhead of moving VAPs between APs. We believe that investigating the pros and cons of these design approaches is necessary, and mechanisms are required to integrate the benefits of these approaches.

AP programmability at layer-1 and layer-2 requires the use of software-defined radio (SDR) platforms such as OpenRadio [129], Atomix [130] or Sora [131]. When integrated into a SDWLAN architecture, using SDRs enables us to offer a richer set of programmability options to network application developers. Additionally, integrating SDRs with SDWLANs simplifies the upgrade of communication standards when the SDR hardware is capable of supporting the new technology. For example, when the digital signal processor (DSP) used on a SDR is fast enough to support new modulation and coding schemes, the controller can update APs to match the capabilities of clients. Furthermore, when signal processing operations are offloaded to a cloud platform, advanced signal processing techniques may be applied to the incoming signals to cope with interference. While SDWLAN-SDR integration is an ongoing research trend, it is also important to study the implications of SDR platforms (e.g., FPGA, DSP, general-purpose processor) on architecture in terms of factors such as control plane delay and overhead.

Although most architectures only rely on the information collected from APs, some architectures, such as Dyson and SplitAP, also require client modification to collect information from the clients' point of view. Extending data plane programmability to clients enables the use of more sophisticated control mechanisms. For example, this would enable the control plane to adjust clients' MAC parameters based on network dynamics and user demands, which also simplifies resource allocation and network slicing. However, in addition to client modification, which may not be desirable for public access networks, extending control plane to clients introduces traffic overhead. Although newly proposed mechanisms such as FlashBack [132] enable low-overhead exchange of control information with clients, the integration of these mechanisms with SDWLANs is not clear. Accordingly, we believe that client programmability has not been well studied in the literature. Additionally, though the 802.11r and 802.11k

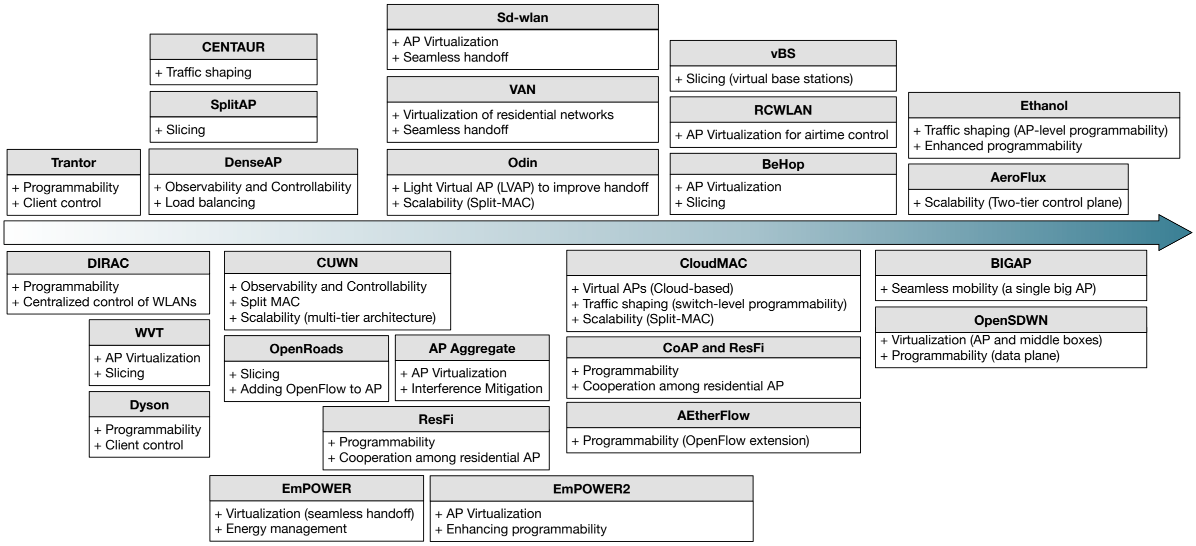
Sd-wlan
+ AP Virtualization
+ Seamless handoff
CENTAUR
+ Traffic shaping
vBS
+ Slicing (virtual base stations)
VAN
+ Virtualization of residential networks
+ Seamless handoff
SplitAP
+ Slicing
RCWLAN
+ AP Virtualization for airtime control
Ethanol
+ Traffic shaping (AP-level programmability)
+ Enhanced programmability
Trantor
+ Programmability
+ Client control
DenseAP
+ Observability and Controllability
+ Load balancing
Odin
+ Light Virtual AP (LVAP) to improve handoff
+ Scalability (Split-MAC)
BeHop
+ AP Virtualization
+ Slicing
AeroFlux
+ Scalability (Two-tier control plane)
DIRAC
+ Programmability
+ Centralized control of WLANs
CUWN
+ Observability and Controllability
+ Split MAC
+ Scalability (multi-tier architecture)
CloudMAC
+ Virtual APs (Cloud-based)
+ Traffic shaping (switch-level programmability)
+ Scalability (Split-MAC)
BIGAP
+ Seamless mobility (a single big AP)
OpenSDWN
+ Virtualization (AP and middle boxes)
+ Programmability (data plane)
WVT
+ AP Virtualization
+ Slicing
OpenRoads
+ Slicing
+ Adding OpenFlow to AP
AP Aggregate
+ AP Virtualization
+ Interference Mitigation
CoAP and ResFi
+ Programmability
+ Cooperation among residential AP
Dyson
+ Programmability
+ Client control
ResFi
+ Programmability
+ Cooperation among residential AP
AEtherFlow
+ Programmability (OpenFlow extension)
EmPOWER
+ Virtualization (seamless handoff)
+ Energy management
EmPOWER2
+ AP Virtualization
+ Enhancing programmability

Fig. 7. This figure summarizes the main features of SDWLAN architectures based on the categorization presented in Section III-C.

amendments (which have been recently integrated into 802.11 standard [133]) realize richer interactions between clients and APs, the SDWLAN architectures did not benefit from these features.

*2) **Mobility**:* Mapping VAPs to physical APs is the main technique used by architectures to support seamless mobility. As each VAP is represented by a simple data structure that holds client's status, exchanging VAPs between APs removes the burden of re-association, as offered by Odin. To avoid the overhead of VAP migration, CloudMAC settles the APs in cloud servers, and programs the switches to route data between physical APs and VAPs. However, performance evaluation of CloudMAC shows that, when traffic is transmitted through a shared network, high-priority packets may be lost or delayed, thereby reducing the chance of connection establishment to 55%. Although queue management has been proposed as a remedy, the effectiveness of such approaches for different topologies, mobility, and traffic patterns, is an open research area. Meanwhile, we believe that the match-action paradigm of OpenFlow must be exploited to improve scalability and handoff performance through designing traffic shaping and prioritization mechanisms that ensure the real-time delivery of time-critical packets. However, the literature does not present any relevant contribution. Another solution is to use network slicing, as we will explain in Section III-D3. On the other hand, most of the reviewed architectures (except CUWN and AeroFlux) use a single-tier controller topology, and unfortunately, they do not present an evaluation of handoff performance in a real-world scenario with tens of APs, background traffic and variable mobility patterns. For example, Odin, ÆtherFlow and BIGAP use testbeds with 10, 2 and 2 APs, respectively. Therefore, a study of architectural implications on handoff performance is missing. We present further discussion about the challenges of mobility management after the review of central control mechanisms in Sections IV-C and V-C.

*3) **Network Virtualization**:* Our review shows that the most common approach of network slicing is through the use of VAPs. For example, in Odin, a slice is defined as a set of APs (Odin Agents), LVAPs corresponding to clients, SSIDs, and network applications. Each SSID may belong to one or multiple slices, and each client joins the slice to which the SSID belongs to. Since each application can only see the LVAPs belonging to its slice, slice isolation is logically provided through the concept of LVAPs.

Although VAP migration can be used for slicing airtime, bandwidth and processing power, it does not provide a fine granularity for resource allocation. For example, if the controller moves a client's VAP to another AP in order to increase its available bandwidth, the actual bandwidth offered depends on the activity of other associated clients. In order to achieve high granularity, layer-1 and layer-2 resource slicing techniques are required to support airtime, frequency and space multiplexing. For example, a middleware should manage the access of VAPs to the AP's physical resources through TDMA scheduling. However, compared to wired networks, the highly variable nature of wireless communications makes predictable slicing a challenging task. Specifically, it is hard to predict and manage the effect of one slice on another. For example, user mobility and variations of traffic in one slice affect the available bandwidth experienced by the users in another slice [113]. Compared to LTE networks, resource slicing is more challenging in 802.11 networks because 802.11 relies on random access mechanisms (i.e., CSMA) and does not utilize a dedicated control plane to communicate with clients. The problem is exacerbated when multiple networks managed by different entities coexist. To cope with these challenges, the existing network slicing techniques heavily rely on a localized network view [96], [134]. For example, while SplitAP supports central network control, each AP decides about its uplink airtime allocation individually and through a distributed algorithm. Based on this discussion, utilizing algorithms that rely on global network view, and the integration of centralized AsC and virtualization are necessary towards improving the capacity and QoS of virtualized networks.

TABLE II
COMPARISON OF SDWLAN ARCHITECTURES

Architecture	Programmable Architecture	Details of Provided APIs	VAP	Network Slicing	South-bound Protocol	Split-MAC Support	Client Modification	Implemented Mechanisms	
								AsC	ChA
DenseAP [12]	×	×	×	×	Proprietary	×	×	✓	×
CUWN [85]	×	×	×	×	CAPWAP	✓	×	✓	✓
DIRAC [88]	✓	✓	×	×	Proprietary	×	×	✓	×
Trantor [89]	✓	✓	×	×	Proprietary	×	✓	×	×
Dyson [90]	✓	✓	×	×	Proprietary	×	✓	✓	×
ÆtherFlow [92]	✓	✓	×	×	Extended OpenFlow	×	×	✓	×
WVT [98]	×	×	✓	✓	Proprietary	×	×	×	×
AP Aggregate [101]	×	×	✓	✓	Proprietary	×	×	✓	×
OpenRoads [99], [100]	✓	✓	×	✓	OpenFlow+SNMP	×	×	✓	×
Odin [49], [102], [103]	✓	✓	✓	✓	OpenFlow+Proprietary	✓	×	✓	×
SplitAP [96]	×	×	✓	✓	Proprietary	×	✓	×	×
CloudMAC [26], [50]	✓	×	✓	✓	OpenFlow	✓	×	✓	✓
EmPOWER [107]	✓	×	✓	✓	OpenFlow+REST	✓	×	✓	×
EmPOWER2 [24]	✓	✓	✓	✓	OpenFlow	✓	×	✓	✓
Sd-wlan [110]	✓	×	✓	×	Extended OpenFlow	✓	×	✓	×
BeHop [112]	✓	×	✓	×	OpenFlow+Proprietary	✓	×	×	×
RCWLAN [113]	×	×	✓	✓	Proprietary	×	×	×	×
vBS [114]	✓	×	✓	✓	OpenFlow	×	×	✓	×
BIGAP [97]	✓	✓	✓	×	Proprietary	×	×	✓	✓
AeroFlux [117]	✓	✓	✓	×	Extended OpenFlow	✓	×	✓	×
Ethanol [121]	✓	✓	✓	×	OpenFlow+Proprietary	✓	×	✓	×
OpenSDWN [25]	✓	✓	✓	✓	OpenFlow+REST	✓	×	✓	×
VAN [105]	✓	✓	×	✓	OpenFlow	×	×	×	×
COAP [31]	✓	✓	×	×	OpenFlow+Proprietary	×	×	×	✓
ResFi [32]	✓	✓	✓	×	Proprietary	×	×	✓	✓

Network slicing can also be used to tackle the challenges of control and data plane communication. To this end, an abstraction layer slices the resources of switching elements based on the communication demands of higher layer control mechanisms. For example, based on the mobility pattern and number of clients, an AsC mechanism may request the abstraction layer for the allocation of switching resources. This is an open research area.

Network slicing would also enable the coexistence of M2M communication with regular user generated traffic, similar to the 5G vision given in [135]. For example, an IoT device may utilize multiple slices of a network to transmit flows with different QoS requirements. Developing such architectures and their associated control mechanisms is an open research area. We present further discussion about this after the review of control mechanisms, in Section V-C4.

*4) **Network Function Virtualization**:* The most common usage of NFV in SDWLANs is split-MAC, which is mainly implemented through VAP. Split-MAC was first introduced by the split-MAC protocol implemented in CAPWAP. As an another example, CloudMAC separates MAC functionalities between physical AP and cloud platform. Using the split-MAC strategy, time critical operations run on APs, while other MAC operations are handled by the controller. This strategy enables exploiting the resources of remote computing platforms and simplifies MAC and security updates without requiring to replace APs as either no or minimal changes to APs is required. Despite the benefits of VAPs, it is important to establish a balance between flexibility and communication overhead because, when more functionalities are shifted to a controller, the overhead and delay of distribution system increase. For example, even for small networks, the importance of utilizing queuing strategies with OpenFlow switches has been demonstrated [50], [121], [136]. Consequently, studies are required to show the tradeoffs between centralization and scalability.

SDR is an another example of NFV. SDR platforms transfer radio signal processing operations into a general-purpose processor on the same board or a remote server. In addition to improving programmability, this enables the use of powerful processors for centralized signal processing, which can be used to improve signal decoding probability and coping with interference challenges. As mentioned in Section III-D1, the integration of SDRs with SDWLANs is in its early stages.

*5) **Home WLANs**:* The importance of utilizing SDN-based mechanisms in home networks is justified given the facts that the number of 802.11 connected devices is being increased (e.g., home appliances, lighting and HVAC, medical monitoring) and some of these devices require timely and reliable data exchange with APs [1], [17], [137]. In addition, the emergence of 802.11 mesh technologies that rely on the installation of multiple APs per home [138] makes central control of home networks even more important. Compared to enterprise networks, the topology of AP placement in home networks is random and variable. Topology is even more randomized when users or distributed algorithms modify the transmission power and channel of their APs. This has serious implications: While ISP-based control (e.g., VAN) simplifies the installation and operation of home networks, the ISP may not be able to collect enough information from the home network because not all the neighbors choose the same ISP. On the other hand, collaboration between homes (e.g., ResFi) introduces serious security challenges. Finally, cloud-based control may not be very effective because neighboring users may not subscribe to the same service. Therefore, the opportunities and challenges of home SDWLANs requires further research.

*6) **Security**:* In addition to running new network control mechanisms such as AsC and ChA, programmability (see Section III-D1) simplifies security provisioning. For example, when a security mechanism does not require hardware replacement (e.g., 802.11i's WPA), it can be implemented by reprogramming APs and clients. However, the decomposition of control and data plane exposes security threats due to enabling of remote programming of network equipment. For example, a DoS attack can be implemented by instructing a large number of clients to associate with an AP. Therefore, it is important to identify security threats and take them into account during the architecture design process.

As mentioned earlier, virtualization through network slicing is an effective approach to control and isolate clients' access to resources. However, as users will be sharing an infrastructure, achieving a strict isolation becomes more challenging. For example, a client may simply switch to the frequency band of another slice to disrupt the capacity offered in that slice. Unfortunately, secure virtualization has not been investigated by the research community.

SDWLANs enable network applications and administrators to detect abnormal activities and network breaches. In the architectures that extend their control plane to clients in order to exchange monitoring and control packets (e.g., Trantor, Dyson, SplitAP), the reports received from malicious clients result in control decisions that negatively affect network performance. To cope with this challenge, a malicious client may be detected using the flow statistics collected from APs and switches [139]. In the next step, client location may be found through analyzing RSSI measurements [140]. Rogue (unauthorized) APs can be detected through collaboration among APs. For example, an abnormal interference level or sniffing the packets being exchanged with a rogue AP can be used by the detection algorithm. Depending on the architecture in use, a network application can block the access of such APs through various mechanisms. For example, using Odin, client access can be blocked through updating OpenFlow switches. Proposing security mechanisms that benefit from the features of SDWLAN architectures is an important future direction, especially for large-scale and public networks.

IV. Centralized Association Control (AsC)

In this section we review centralized AsC mechanisms. We focus on the metrics employed as well as the problem formulation and solving approaches proposed. This section also highlights the impact of architecture on design when details are available. We employ a consistent notation to ease the understanding and comparison of metrics and formulations. Our review summarizes the performance improvements achieved to reveal the benefits of these centralized mechanisms compared to distributed approaches.[2] Although we mostly focus on state-of-the-art centralized mechanisms, we review the seminal distributed mechanisms as well because of their adoption as the baseline to evaluate the performance of centralized mechanisms.

At a high level, we categorize AsC mechanisms from two perspectives:

- ***Seamless handoff***: refers to the mechanisms that their objective is to reduce the overhead and delay of client handoff,
- ***Client steering***: refers to the mechanisms that adjust client-AP associations to optimize parameters such as the load of APs and the airtime allocated to clients.

Supporting seamless handoff is usually addressed by proposing architectures that reduce the overhead of re-association. In contrast, client steering is performed through proposing AsC mechanisms that run on the control plane. For example, an AsC mechanism may propose an optimization problem to balance the load of APs, while handoff delays depend on the architectural properties. From the client steering point of view, AsC is particularly important in dense topologies because there are usually multiple candidate APs for a client in a given location. Therefore, using a simple RSSI metric may result in hot spots, unbalanced load of APs and unfair resource allocation to clients. Hence, one of the main goals of AsC is to achieve *fairness* among clients and APs.

The distributed AsC mechanism employed by 802.11 standard is **strongest signal first (SSF)** [141], [142]. Using SSF, each client decides about its association based on the RSSI of probe response and beacon messages. Each client associates with the AP from which highest RSSI has been received. **Least load first (LLF)** [143] is an another widely-adopted distributed mechanism where APs broadcast their current load through beacon messages to help the clients include AP load when making association decisions. The load of an AP is represented through various metrics, such as the number of associated clients[3]. In the following, we review AsC mechanisms.

A. Seamless Handoff

Some of the architectures reviewed in Section III-C propose mechanisms to reduce the overhead of client handoff. Handoff overhead refers to: (i) the packets exchanged between client and AP to establish a connection, and (ii) the delay incurred by the client during this process [144]. In this section we study the contributions of SDWLAN architectures in terms of supporting seamless handoff.

Cisco unified wireless network (CUWN). [85] enables the central configuration of RSSI threshold with hysteresis to perform seamless handoff of CCX [145] compatible clients. When the RSSI received from associated AP drops below the *scan threshold*, the client increases its AP scanning rate to ensure fast handoff to another AP when the difference between the RSSI of the associated and new AP is equal or greater than the *hysteresis* value specified. In addition to hysteresis, specifying the *minimum RSSI* value forces the clients to re-associate when their RSSI drops below a minimum RSSI.

Three types of client roaming scenarios are handled by CUWN: (i) intra-controller roaming, (ii) inter-controller layer-2 roaming, and (iii) inter-controller layer-3 roaming. The controller simply handles an intra-controller roaming by updating its client database with the new AP connected to

[2] Note that in this section and the next section (Section V) we do not study all the AsC and ChA mechanisms implemented by the architectures reviewed in Section III. This is because some of these architectures only show the feasibility of implementing control mechanisms, and they do not propose any *new* mechanism to benefit from the features of SDWLANs.

[3] The traffic indication map (TIM) of a beacon packet represents a bitmap that indicates the clients for which the AP has buffered packets.

the roaming client. Inter-controller layer-2 roaming occurs when a client associates with an AP that is controlled by a different controller belonging to the same subnet. In this type of roaming, mobility messages are exchanged between the old controller and the new one. Then, the database entry related to the roaming client is moved to the new controller. Inter-controller layer-3 roaming occurs when a client is associated with an AP that is controlled by a controller belonging to a different subnet. In layer-3 roaming, the database entry of the roaming client is not moved to the new controller. Rather, the old controller marks the client's entry as *anchor entry*. The entry is copied to the new controller and marked as *foreign entry*. Therefore, the original IP address of the roaming client is maintained by the old controller. CUWN enables network administrators to establish mobility groups, where each group may consist of up to 24 controllers. A client can roam among all the controllers in a mobility group without IP address change, which makes seamless and fast roaming possible.

Odin's Mobility Manager (OMM). Odin [49], [102] enables seamless handoffs through LVAP migration between APs. The Odin controller maintains a persistent TCP connection per Odin Agent running on AP, thereby, switching among agents does not require connection reestablishment. In addition, the delay of a re-association equals the delay of sending two messages from the Odin controller to the old and new APs. The first message removes an LVAP from the old AP, and the second message adds an LVAP to the new AP. Assuming that the messages are sent successfully, the delay equals the longest round trip time (RTT) between the controller and the two APs, which depends on network size.

To show the effectiveness of LVAPs, Odin employs a simple RSSI-based AsC mechanism. The mobility application (running on the controller) selects the Odin Agent with highest RSSI if client movement is detected by the controller. The effect of handoff on TCP throughput has been evaluated using two APs and a client. While a period of throughput drop in regular 802.11 layer-2 handoff is observable, Odin does not show any throughput degradation. Performance evaluations also show that executing 10 handoffs per second results in a negligible reduction in TCP throughput.

ÆtherFlow. [92] argues that handoff support through LVAP migration (e.g., Odin [102] and OpenSDWN [25]) imposes high computational and communication overhead, especially in large networks with many mobile clients. They propose a predictive handoff strategy by relying on the extended OpenFlow protocol proposed in this work (see Section III-C2). The handoff mechanism works in three phases:

- *Prediction*: The controller predicts an association when the RSSI of a client to its associated AP declines while its RSSI to another AP increases.
- *Multicasting*: By updating OpenFlow tables, the controller multicasts the packets of the client to both the current AP and the predicted AP.
- *Redirection*: The controller will redirect the client's traffic to the new AP after the handoff completion. If no handoff occurs, then the multicasting is stopped.

Experiments using two APs and a client shows that the handoff delay of ÆtherFlow is around 5.3 seconds, compared to the 7.1 seconds delay of 802.11 standard.

BIGAP. [97] uses the 25 non-overlapping channels of 5GHz band to form disjoint collision domains for handoff. As explained in Section III-C, a separate NIC is used to periodically overhear packets on all channels, which enables the controller to compute the potential SNR values of client-AP links. A handoff happens when a higher SNR would be achievable for a client. However, to avoid the ping-pong effect, an 8dB hysteresis value is used.

BIGAP performs handoff through client channel switching. Since all APs share the same BSSID, in order to handoff a client from AP_1 to AP_2, the controller instructs AP_1 to send a channel switching command to the client. The client then switches to the channel being used by AP_2. Since both APs use the same BSSID, the client does not notice handoff.

Performance evaluations (using two APs) show that the BIGAP handoff is about 32 times shorter than the regular 802.11 handoff. In addition, BIGAP results in lower energy consumption because, it moves the overhead of handoff to APs and there is no need for the clients to scan the channels. In addition, while 802.11 results in zero throughput for about 4 seconds, BIGAP shows only 5% throughput reduction during the handoff.

B. Client Steering

Based on the scope of the optimization problem employed, we categorize client steering mechanisms into two groups: ***centrally-generated hints*** and ***centrally-made decisions***. In AsC mechanisms using centrally-generated hints, the controller relies on the global network view to generate hints for the association of clients. In other words, the controller does not make the final decision about associations and instead, enables the clients to make more informed decisions through the hints conveyed. On the other hand, in AsC mechanisms using centrally-made decisions, the controller makes the association decisions and enforces the clients to apply them. We will discuss in Sections IV-B2 and IV-B3 the two sub-categories of mechanisms based on centrally-made decisions.

*1) **Centrally-Generated Hints**:* In this section we review AsC mechanisms that employ client steering through hints generated centrally.

BestAP. [146] proposes an AsC mechanism based on the estimation of *available bandwidth* (ABW) for each client at every AP in its vicinity. The available bandwidth depends on channel load, which varies with packet loss and PHY rate. The estimated available bandwidth at PHY rate r is computed as,

$$E[ABW(r)] = \frac{8S_{data}(1-B)}{\sum_{k=0}^{n}(1-p_s(r))^k p_s(r) T_{tx}(r,k)}, \qquad (1)$$

where S_{data} is data size (bytes), p_s is the probability of successful packet transmission, $T_{tx}(r,k)$ is the transmission time of a packet during the kth transmission attempt with PHY rate r, and B is the fraction of channel busy time. B is measured through using the CCA register of NIC. $p_s(r)$ is measured using the statistics provided by the rate adaptation algorithm. All other parameters are configured statically.

A scheduler is run on each client to allocate a measurement period (e.g., 50ms every 2s) during which the client sends data to all nearby APs in order to update packet loss and channel busy fraction. A monitoring service is run on APs to collect the statistics from clients, calculate ABW, and send a report to the controller. The controller sends the estimated ABWs of the best five APs to each client periodically. BEST-AP only considers the ABW of downlinks.

Testbed evaluations show that the delay overhead of ABW estimation is less than 50ms, which is appropriate for a dynamic AsC mechanism. Compared to SSF [141], the proposed AsC mechanism shows 81% and 176% improvement in throughput for static and mobile clients, respectively.

Ethanol. This architecture [121] (see Section III-C5) has been evaluated through running a load-aware AsC mechanism that aims to balance the number of associated clients among APs. When a client requests to join an AP with higher load, the controller drops the request to force the client look for another AP. A simple testbed with two APs and up to 120 clients shows that the maximum difference between the number of clients associated to APs is two, which is due to the concurrent arrival of association requests.

*2) **Individual Optimization through Centrally-Made Decisions**:* In this section we review AsC mechanisms that employ client steering through decisions generated centrally. These mechanisms, however, do not define a global optimization problem; thereby they do not take into account the effect of an association on other clients/APs. Due to the individual nature of association control, these mechanisms only improve the overall network performance, and cannot be used to enforce fairness.

DenseAP. The AsC mechanism proposed by DenseAP [12] (see Section III-C1) works as follows. The *available capacity* metric is defined to rank all the APs a client could be associated with. A client associates to the AP with highest available capacity. The available capacity of AP_i operating on channel ch_i is defined as follows,

$$AC^{ch_i}_{AP_i,c_j} = F^{ch_i}_{AP_i} \times r^{ch_i}_{AP_i,c_j}, \qquad (2)$$

where $F^{ch_i}_{AP_i}$ is the *free airtime* of AP_i on channel ch_i, and $r^{ch_i}_{AP_i,c_j}$ is the *expected transmission rate* of c_j when communicating with AP_i. Free airtime is estimated by measuring the amount of time that the MAC layer contends for channel access to send a high-priority packet. The expected transmission rate is estimated using the RSSI of probe request frames received at the AP. A mapping table is used for this purpose. APs hear the probe requests of clients and send reports to a controller. APs also measure their free airtime and report it to the controller. The controller selects the AP with highest available capacity for each client to associate with. To instruct a client associate with the selected AP, only the selected AP responds to the client's probe message.

DenseAP also proposes a dynamic load balancing algorithm that periodically decides about associations. In particular, the controller checks the free airtime of APs every minute. An AP is *overloaded* if its free airtime is less than 20% and it has at least one associated client. If an overloaded AP exists, the controller considers its clients as the potential candidates for association with APs experiencing lower load. A candidate AP must satisfy these conditions: (i) the expected transmission rate of clients when associated with the new AP must not be lower than the current transmission rate, and (ii) the free airtime of new AP must be at least 25% more than the current AP. During each decision period, at most one client is allowed to be associated with a new AP, and two consecutive associations for a client is prohibited to avoid the ping-pong effect.

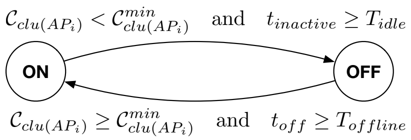
$\mathcal{C}_{clu(AP_i)} < \mathcal{C}^{min}_{clu(AP_i)}$ and $t_{inactive} \geq T_{idle}$
ON
OFF
$\mathcal{C}_{clu(AP_i)} \geq \mathcal{C}^{min}_{clu(AP_i)}$ and $t_{off} \geq T_{offline}$

Fig. 8. Finite state machine (FSM) of energy manager for a slave AP in EmPOWER [107].

Empirical evaluations show a 40% to 70% increase in per-client throughput, compared to SSF. Moreover, the authors conduct a small experiment using three clients and two APs to show that the load balancing algorithm improves the throughput of clients by more than 200%.

Odin's Load Balancing (OLB). This load-balancing mechanism [102] periodically (every minute) inquires APs to collect the RSSI relationship between clients and APs, and then LVAPs are evenly distributed between APs to balance their loads. The evaluations on a testbed with 10 APs and 32 clients show that around 50% and 15% of clients were able to receive a fair amount of throughput when the proposed mechanism was enabled and disabled, respectively.

EmPOWER. This mechanism [24], [107] relies on client steering to improve the energy efficiency of APs. APs are partitioned into clusters, where each cluster has one master and multiple slaves. Master APs are always active, and they are manually selected during the network deployment phase to provide a full coverage. Slave APs are deployed to increase network capacity. These APs are turned on/off using the finite state machine (FSM) depicted in Figure 8. In the ON mode, all wireless interfaces of the AP are on. In OFF mode, only the Energino [108] module of the AP is on.

Two metrics are defined for a slave AP_i belonging to cluster $clu(AP_i)$: $\mathcal{C}_{clu(AP_i)}$ is the number of clients in the cluster, and $\mathcal{C}^{min}_{clu(AP_i)}$ is the minimum required number of clients in cluster $clu(AP_i)$ to keep AP_i active. A slave AP_i transitions from ON mode to OFF mode if: (i) the number of its cluster's clients is less than $\mathcal{C}^{min}_{clu(AP_i)}$, and (ii) AP_i has been inactive (i.e., no client associated) for at least T_{idle} seconds. Also, AP_i transitions from OFF mode to ON mode if: (i) the number of its cluster's clients is at least $\mathcal{C}^{min}_{clu(AP_i)}$, and (ii) AP_i has been OFF for at least $T_{offline}$ seconds.

The mobility manager associates a client to a new AP with higher SNR. However, to establish a balance between performance and energy consumption, the mobility manager may associate a client to an AP with lower SNR but smaller $\mathcal{C}^{min}_{clu(AP_i)}$. In this way, the energy manager is able to turn off the APs with higher $\mathcal{C}^{min}_{clu(AP_i)}$ in order to decrease energy consumption. Re-association is performed if there is a better AP in terms of SNR and $\mathcal{C}^{min}_{clu(AP_i)}$. By relying on the Odin [102] APIs, the authors showed that this AsC mechanism was implemented as a Java network application with only 120 lines of codes.

Adaptive mobility control (AMC). [147] shows that using a fixed RSSI threshold by clients results in an unbalanced load of APs. They propose a heuristic algorithm which uses RSSI and traffic load of APs to provide dynamic hysteresis margins on AP traffic load level. The load of AP_i is defined as,

$$L_{AP_i} = \begin{cases} B & \text{if } |\mathcal{C}_{AP_i}| = 0 \\ 0.8 \times B + 0.2 \times |\mathcal{C}_{AP_i}| & \text{if } |\mathcal{C}_{AP_i}| > 0 \end{cases} \tag{3}$$

where $|\mathcal{C}_{AP_i}|$ is the number of clients associated with AP_i, and B is channel busy time. The algorithm defines three thresholds on AP load and RSSI: low, medium, and high. Using these thresholds, a client is associated with a new AP that satisfies one of these conditions: (i) higher signal strength and lower load, (ii) significantly higher signal strength and slightly higher load, or (iii) significantly lower load and slightly lower signal strength.

The proposed adaptive mobility manager has been implemented using Odin. Empirical results show more than 200% improvement in TCP throughput, compared to SSF.

Associating to good enterprise APs (AGE). The main objective of AGE [148] is the reduction of clients' packet exchange delay over wireless links through client steering. AGE has two main phases: learning and AP selection. During the learning phase, the performance metrics and environmental factors are pulled every minute from all APs using SNMP [60]. These metrics include AP-client RTT, RSSI, SNR, number of associated clients, channel number, frequency band, AP location, day of week, and time.

Using the collected training set, the authors use the *random forest* [149] technique to generate a two-class learning model for classifying APs into *high latency* and *low latency*. Figure 9 shows an overview of AGE and its two main components: (i) the *AGE app* that is installed on each client's mobile phone, and (ii) the *AGE controller* that uses the random forest model to classify the APs in the vicinity of each client. AGE operates as follows:

- The AGE application on a client device sends an AGE request (including the list of achievable APs) to the AGE controller, periodically.
- The AGE controller pulls the SNMP data of the APs requested by AGE app.
- The latency class of each AP is predicted using the learning technique mentioned earlier.
- The AGE controller informs the AGE application about the best AP nearby.
- The client re-associates with a new AP using AGE app.

The authors deployed AGE at Tsinghua University campus where over 1000 devices used the network for 2.5 months. Measurements confirmed that the wireless data exchange delay of more than 72% of clients has been reduced by over 50%.

*3) **Global Optimization through Centrally-Made Decisions:*** In this section we review AsC mechanisms that employ client steering by formulating a problem that aims to optimize performance parameters globally. These mechanisms propose heuristics to solve the NP-hard problems that usually aim to achieve network-wide fairness.

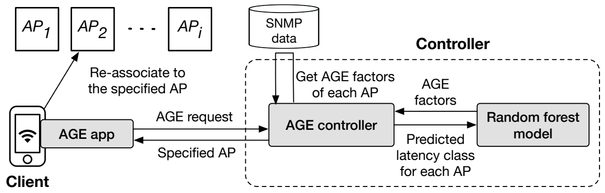

Fig. 9. Associating to Good Enterprise APs (AGE) [148]. Each client is instructed to connect to the AP that provides minimum delay.

Association Control for fairness and load balancing (ACFL). In [150] and [151], the authors address the unbalanced load of APs as a result of using SSF or LLF. To address this challenge, an AsC problem is formulated to establish the max-min bandwidth fairness among APs. Intuitively, the load of a client on its associated AP is inversely proportional to the effective bit rate of AP-client link. The load of AP_i (denoted as L_{AP_i}) is modeled as the maximum of the aggregated loads of its wireless and wired links generated by all clients $c \in \mathcal{C}$, as follows,

$$L_{AP_i} = \max\left\{ \sum_{\forall c_j \in \mathcal{C}} \frac{\omega_{c_j} \times X_{AP_i,c_j}}{r_{AP_i,c_j}}, \sum_{\forall c_j \in \mathcal{C}} \frac{\omega_{c_j} \times X_{AP_i,c_j}}{R_{AP_i}} \right\} \tag{4}$$

where r_{AP_i,c_j} is the transmission rate between AP_i and c_j, R_{AP_i} is the transmission rate of wired interfaces of AP_i, $X_{AP_i,c_j} \in \{0,1\}$ is the association state of c_j to AP_i, and ω_{c_j} is the traffic volume size of client c_j. In other words, the load of AP_i is defined as the period of time this AP requires to handle the traffic of its associated clients. AP_i provides bandwidth $X_{AP_i,c_j} \times \omega_{c_j}/L_{AP_i}$ to client c_j.

Two approximation algorithms are introduced to solve the formulated NP-hard max-min fair bandwidth allocation problem. The first algorithm solves the problem for unweighted greedy clients. The second algorithm proposes a solution for weighted and limited throughput demand of clients. The algorithms are run periodically by a controller to update client associations. Simulation results show over 20% improvement in terms of average per-client bandwidth, compared to SSF and LLF.

Association control for proportional fairness (ACPF). [152] argues that using throughput-based fairness (e.g., [153], [154]) in multirate networks leads to low overall network throughput because clients with a low bit rate can occupy the channel longer than those with higher rate. The authors investigate the problem of achieving proportional fairness by introducing the following objective function formulated based on the effective bandwidth of clients:

$$\sum_{\forall c_j \in \mathcal{C}} \rho_{c_j} \times \log(\beta_{c_j}), \tag{5}$$

where ρ_{c_j} and β_{c_j} are the priority and effective bandwidth of client c_j, respectively. β_{c_j} is defined as

$$\sum_{\forall AP_i \in \mathcal{AP}} X_{AP_i,c_j} \times t_{AP_i,c_j} \times r_{AP_i,c_j}, \tag{6}$$

where $X_{AP_i,c_j} \in \{0,1\}$ is the association index, $t_{AP_i,c_j} \in [0,1]$ is the effective normalized time, and r_{AP_i,c_j} is the

PHY rate between c_j and AP_i. Achieving fairness through association is formulated as a linear-programming problem to maximize the objective function. A centralized algorithm, called *non-linear approximation optimization for proportional fairness* (NLAO-PF), is proposed to solve the optimization problem. It has been proven that the approximation ratio of NLAO-PF is 50%.

The authors used the OMNet++ [155] simulator to evaluate the effectiveness of NLAO-PF compared to cvapPF [156] in terms of average throughput. The average improvement is reported as 18.8% and 35% for uniformly distributed clients as well as clients in a hotspot area, respectively.

Association control with heterogeneous clients (ACHC). Experimental analysis of the negative impacts of legacy clients (802.11a/b/g) on 802.11n clients has been reported by [157]. The authors propose a two-dimensional Markov model to calculate the uplink and downlink throughput of heterogeneous clients. MAC efficiency of a client c_j is defined as follows,

$$\Upsilon(c_j) = \frac{Th_{c_j}^{up} + Th_{c_j}^{down}}{\min\{1,\ p_{c_j}^{up} + p_{c_j}^{down}\} \times r_{AP_i,c_j}}, \qquad (7)$$

where $Th_{c_j}^{up}$ and $Th_{c_j}^{down}$ are the estimated uplink and downlink throughput of client c_j, respectively; $p_{c_j}^{up}$ and $p_{c_j}^{down}$ are, respectively, the uplink and downlink traffic probabilities of c_j; r_{AP_i,c_j} is the optimal PHY rate between c_j and AP_i. AsC is formulated through an optimization problem to provide a bandwidth proportional to each client's achievable data rate, which is obtained by maximizing $\sum_{\forall c_j \in \mathcal{C}} \log \Upsilon(c_j)$.

Two heuristic dynamic AsC algorithms are proposed to solve the optimization problem: *FAir MAC Efficiency* (FAME) and *Categorized*. Instead of maximizing all MAC efficiencies, these algorithms maximize the minimum MAC efficiency. To run FAME, the data rate and traffic load of all clients in nearby BSSs must be collected from each client's point of view. With *Categorized*, the type of all clients in nearby BSSs must be collected as well. However, *Categorized* is less sensitive to network dynamics because it tries to maximize the number of similar-standard 802.11 clients connected to each AP. Therefore, *Categorized* takes advantage of dense deployment and alleviates the performance degradation of 802.11n clients that is caused by the presence of legacy clients.

Simulation and testbed experiments confirmed the higher performance of *FAME* and *Categorized* compared to SSF and ACFL in terms of TCP and UDP throughput, MAC efficiency and aggregated throughput of clients supporting different standards. Although the conducted experiments use distributed execution of *FAME* and *Categorized*, collecting the information required by these algorithms imposes a high burden on clients. Therefore, these algorithms are mostly suitable for SDWLANs.

Association control and CCA adjustment (ACCA). Two central algorithms for AsC and CCA threshold adjustment are proposed in [158] for dense AP deployments. The basic idea of the AsC mechanism is to utilize the SINR of a client perceived by an AP in order to measure the interference (congestion) level of APs. The network is modeled as a weighted bipartite graph (WBG), and the AsC problem is formulated as a maximum WBG matching combinatorial optimization problem. The weights of edges in WBG are the uplink SINR values. The optimization problem's objective is to maximize the sum of the clients' throughput, where the throughput upper-bound is computed using the Shannon-Hartley formula [159]. Per-BSS CCA level is adjusted by using a constraint of the optimization problem based on the cell-edge SINR of each BSS. The optimization problem has been solved using Kuhn-Munkres [160] assignment algorithm.

MATLAB simulations show a 15 to 58% improvement at the 10th percentile of cumulative distribution of clients throughput, compared to SSF. Furthermore, the CCA adjustment algorithm significantly improves the throughput of cell-edge clients, compared to the network with fixed CCA threshold.

Demand-aware load balancing (DALB). [161] formulates joint AP association and bandwidth allocation as a mixed-integer nonlinear programming problem that includes the bandwidth demand of clients. The objective is to maximize the aggregated bandwidth of clients, which establishes a trade-off between throughput gain and time-based user fairness. The key idea of this work is the inclusion of clients' bandwidth demands in the computation of APs' load. For a client c_j, bandwidth demand is defined as $\beta_{c_j} = S_{c_j}/T$, where S_{c_j} is the size of data received for client c_j by its associated AP during the allocable transmission time of the AP, i.e., T. The transmission time demand of c_j associated to AP_i is defined as $T_{AP_i,c_j} = \beta_{c_j} \times T/r_{AP_i,c_j}$.

The optimization problem's objective is defined as follows,

$$\sum_{\forall c_j \in \mathcal{C}} \log \sum_{\forall AP_i \in \mathcal{AP}} \frac{X_{AP_i,c_j} \times r_{AP_i,c_j} \times t_{AP_i,c_j}}{T}, \qquad (8)$$

where the transmission time allocated to c_j associated with AP_i is expressed by t_{AP_i,c_j}. The authors proved that maximizing Equation 8 subject to a constraint on the transmission time demand of clients, i.e., $0 \leq t_{AP_i,c_j} \leq T_{AP_i,c_j}$, leads to proportional time fairness among clients.

A two-step heuristic algorithm is introduced to solve the optimization problem. The first step associates a client to an AP with the lowest allocated transmission time at each iteration until all clients are associated. At each iteration, the algorithm selects a client with the largest bandwidth demand. The second step schedules the transmission time of APs to establish proportional transmission time fairness among clients.

Simulations show a 23% improvement in clients' throughput compared to SSF, ACPF and NLB [162]. In addition, this improvement has been achieved without sacrificing fairness. In particular, while bandwidth fairness is slightly better than that of SSF, ACPF and NLB, time fairness is significantly improved compared to these algorithms.

Migration-cost-aware association control (MCAC). [163] investigates the problem of max-min fairness subject to a migration cost constraint. An integer linear programming problem is formulated with the aim of maximizing the minimum throughput of clients in order to establish max-min fairness. The migration cost of clients is the main constraint of the formulation. The load of AP_i is defined as $L_{AP_i} = \sum_{\forall c_j \in \mathcal{C}} X_{AP_i,c_j}/r_{AP_i,c_j}$. Assuming the achievable throughput of clients is proportional to the inverse of the load of the associated AP (i.e., $1/L_{AP_i}$), maximizing the minimum throughput of clients is equivalent to minimizing $\max_{\forall AP_i \in \mathcal{AP}} L_{AP_i}$. The migration cost constraint is defined as,

$$\sum_{\forall AP_i \in \mathcal{AP}} \sum_{\forall AP_l \in \mathcal{AP}} (Y_{AP_i,AP_l} \times M_{c_j}) \leq M, \qquad (9)$$

where $Y_{AP_i,AP_l} \in \{0,1\}$ indicates whether a client c_j currently associated with AP_i will be associated with AP_l, M_{c_j} is the migration cost of client c_j from AP_i to AP_l, and M is the total permissible migration cost.

The authors showed that the proposed problem is NP-hard, and they proposed an approximation algorithm, named the *cost-constrained association control algorithm* (CACA), to solve the optimization problem. In this algorithm, the problem is divided into two sub-problems: client removal and client re-association. In the first sub-problem, a subset of clients are removed from their current AP with the aim of minimizing the maximum AP load, considering the migration cost constraint. In the second sub-problem, the algorithm performs client re-association to minimize the load of maximum-loaded AP.

NS3 [164] simulations and experimental results show improvements in terms of the number of re-associations, throughput, loss ratio and end-to-end delay, compared to SSF, ACPF and GameBased [165].

QoS-driven association control (QoSAC). [166] investigates the problem of flow-level association to address the QoS demands of clients. Specifically, the backhaul capacity of APs is included in the association decision making. This work proposes a mechanism for concurrent association of a client to multiple APs and supports flow-level routing of traffic. This mechanism facilitates dedicated management of each flow. For example, a client may use AP_1 for video streaming while using AP_2 to upload a file.

The main objective is to minimize the *average inter-packet delay* of individual download flows. The inter-packet delay for a client associated to an AP is formulated based on the bi-dimensional unsaturated Markov model proposed in [157]. An optimization problem is formulated to minimize the sum of the average inter-packet delay of all APs. The backhaul capacity of an AP should not be smaller than the sum of the arrival traffic rates of the AP, i.e., the sum of all download flow rates. This is included as a constraint of the optimization problem. The proposed optimization problem is interpreted into a supermodular set function optimization [167], which is an NP-hard problem. This problem has been solved using two heuristics: (i) greedy association, and (ii) bounded local search association. The greedy algorithm associates a client with the AP that minimizes the total inter-packet delay in each iteration. In this algorithm, the association loop is continued until all clients are associated. The bounded local search is a polynomial-time algorithm similar to the algorithm proposed in [168].

Simulation results confirm the reduced average inter-packet delay achieved with this mechanism, compared to SSF and FAME [157]. The delay is considerably reduced when the backhaul capacity is limited and clients form hotspots.

C. Association Control: Learned Lessons, Comparison, and Open Problems

Table III presents and compares the features of the AsC mechanisms. Although most of the AsC mechanisms focus on client steering, we should note that client steering and seamless handoff are interdependent. For example, when association decisions are made centrally to balance the load of APs, a quick reaction to network dynamics requires seamless handoff; otherwise, the effect of load balancing will be compromised by the overhead and delay of re-associations. In the following, we study the features of AsC mechanisms and identify research directions.

1) ***Optimization Scope of Client Steering****:* Based on our review, we have classified the client steering approaches into three categories: (i) centrally-generated hints, (ii) individual optimization through centrally-made decisions, and (iii) global optimization through centrally-made decisions. Our review shows that most of the AsC mechanisms fall into the last two categories, where their optimization problems is classified as follows:

– *Overall network performance*: To improve the overall performance of clients and APs. For example, QoSAC minimizes the sum of inter-packet delay for all APs, and ACCA maximizes the sum of clients' achievable throughput. Note that improving overall performance does not necessarily require defining an optimization problem, as Section IV-B2 shows.

– *Max-min fairness*: To maximize the minimum performance. In other words, *max-min* fairness indicates providing a client with more resources would not be possible without sacrificing the resources of other clients [169]. For example, the objective function of ACFL and MCAC maximizes the throughput of minimum-throughput client.

– *Proportional fairness*: Clients occupy the channel proportional to their transmission rate. Our review shows that proposing these approaches is motivated by the challenges of achieving fairness in multi-rate networks. Multi-rate is caused by both client heterogeneity (e.g., 802.11a/b/n) and the rate adaptation mechanism of 802.11 standards. Since the 802.11 DCF protocol aims to provide an equal channel access probability for all contending clients, low-rate clients occupy the medium longer than high-rate clients. In this case, the max-min throughput fairness results in sacrificing the performance of high-rate clients, which leads to a problem called *performance anomaly*. Performance anomaly in multi-rate networks may result in total throughput degradation, even in comparison to SSF [141], [157], [170], [171]. A well-known solution is proportional fairness, which is usually achieved by defining a logarithmic objective function and scheduling the transmission time of clients to be proportional to their transmission rate. ACPF, ACHC and DALB are examples of AsC mechanisms that establish proportional fairness.

Using proportional fairness in multi-rate networks decreases the performance of low-rate legacy clients and may dissatisfy their QoS requirements. We suggest two approaches to ad-

TABLE III
COMPARISON OF ASSOCIATION CONTROL (ASC) MECHANISMS

Mechanism	Objective		Optimization Scope			Decision Metric	Traffic Awareness		Performance Evaluation	
	Seamless Handoff	**Client Steering**	**Cent. Gen. Hints**	**Indv. Opt. Cent. Made Decisions**	**Glob. Opt. Cent. Made Decisions**		**Downlink**	**Uplink**	**Simulation**	**Testbed**
CUWN [85]	✓	×	×		×	RSSI	×	×	×	×
OMM [102]	✓	×	×		×	RSSI	×	×	×	✓
ÆtherFlow [92]	✓	×	×		×	RSSI	×	×	×	✓
BIGAP [97]	✓	×	×		×	RSSI	×	×	×	✓
BestAP [146]	×	Bandwidth Improvement	✓		×	Available bandwidth	✓	×	×	✓
Ethanol [121]	×	AP Load Balancing	✓		×	Number of clients	×	×	×	✓
DenseAP [12]	×	AP Load Balancing	×	Overall performance	×	Available bandwidth	✓	×	×	✓
OLB [102]	×	AP Load Balancing	×	Overall performance	×	RSSI, Number of clients	×	×	×	✓
EmPOWER [107]	×	Energy Efficiency	×	Overall performance	×	RSSI, Number of clients	×	×	×	✓
AMC [147]	×	AP Load Balancing	×	Overall performance	×	RSSI, Free airtime	✓	×	×	✓
AGE [148]	×	Reducing Packet Exchange Delay	×	Overall performance	×	Latency	×	×	×	✓
ACFL [151]	×	AP Load Balancing	×		Max-min fairness	Free airtime, Transmission rate	✓	×	✓	×
ACPF [152]	×	Fair Bandwidth Allocation	×		Proportional Fairness	Available bandwidth	✓	×	✓	×
ACHC [157]	×	Fair Bandwidth Allocation	×		Proportional Fairness	MAC efficiency	✓	✓	✓	✓
ACCA [158]	×	Maximize Throughput	×		Overall Performance	Uplink SINR	×	✓	✓	×
DALB [161]	×	Maximize Throughput	×		Proportional Fairness	Transmission time	✓	×	✓	×
MCAC [163]	×	Fair Bandwidth Allocation	×		Max-min Fairness	Throughput, Re-association cost	✓	×	✓	✓
QoSAC [166]	×	Reducing Packet Exchange Delay	×		Overall Performance	Inter-packet delay	✓	×	✓	×

dress this problem: (i) hybrid fairness, and (ii) demand-aware proportional fairness. In hybrid fairness, a weighted fairness metric is defined by assigning weights for the max-min and proportional fairness metrics. Therefore, it is possible to adjust the weights based on the demand of low-rate legacy clients. In the second approach, the demand of clients is merged with their rates when formulating the problem.

2) ***Decision Metrics:*** The most important component of an AsC mechanism is its decision metrics, as they affect on both accuracy and overhead. When using the AsC mechanism of 802.11 standard (i.e., SSF), a client associates with the AP that is providing the highest RSSI. The main shortcoming of SSF is that it does not recognize the load of APs and the demand of clients. Therefore, AsC mechanisms introduce additional metrics in their decision process.

The ”Decision Metric” column of Table III summarizes the decision metrics employed. Generally, our review shows that the most popular decision metrics are RSSI, AP load, throughput of clients, and packet exchange delay experienced by clients. Balancing the load of APs is a widely-adopted optimization metric as it implicitly results in an improved performance of clients. In the mechanisms relying on this metric, load is modeled through parameters such as the number of associated clients, AP throughput, and channel busy time. Although these mechanisms implicitly improve the performance of clients, explicit consideration of clients’ traffic results in higher performance.

Based on this, we can classify AsC mechanisms into two groups:

– *Demand-agnostic.* Refers to the AsC mechanisms that associate each client with the best AP nearby, without taking into account clients’ demands. For instance, DenseAP, ACPF and BestAP associate each client with the nearby AP that is providing the highest available bandwidth, independent of the current demand of clients. Similarly, AGE and MCAC associate each client with the AP that is providing the minimum delay, without considering the tolerable delay of clients in the decision making process. As a result, demand-agnostic approaches may perform unnecessary associations, which incur handoff overheads that result in performance degradation.

– *Demand-aware.* Refers to the AsC mechanisms that explicitly include the demand of clients in their decision making process. Unfortunately, the number of demand-aware AsC mechanisms (e.g., ACHC, and DALB) is very limited.

Throughput demand, in particular, is composed of uplink and downlink traffic. Unfortunately, most of the AsC mechanisms assume downlink traffic is dominant when calculating their association decision metrics, as Table III shows. However, the emerging applications of SDWLANs justify the importance of uplink traffic. For example, medical monitoring, industrial process control, surveillance cameras, and interaction with cloud storage, require timely and reliable uplink communication [17], [35], [54], [172]–[174].

These discussions reveal that designing AsC mechanisms to support both uplink and downlink requirements is an open problem. Additionally, due to the resource-constraint nature of IoT devices, it is important to define, compute and include metrics, such as channel contention intensity and packet loss rate, in the decision making process.

In addition to the mentioned challenges, it is also important to identify and address the effect of hardware on the metrics used. For example, the RSSI perceived by a tiny IoT device would be different from the value perceived by a multi-antenna smartphone, under the same condition. Therefore, the controller cannot generate a realistic network map based on the information collected from clients. Similarly, for environments with highly asymmetric links (for example caused by shadowing), the information collected in the controller based on the RSSI perceived at the APs does not reflect the real connectivity and interference relationship. Understating the effect of metric evaluation irregularity on performance and the calibration of metrics and inclusion in the AsC mechanisms are open research areas.

*3) **Dynamicity and Overhead**:* Dynamic and scalable AsC requires minimizing overhead from the following point of views:

– *Measurement delay*: the delay of measuring decision metrics must be short to reflect network dynamics;

– *Bandwidth requirements*: the amount of bandwidth consumed for exchanging control data should be minimized to enhance scalability. Each association command requires communication between the controller and APs. In addition, in a VAP-based architecture such as Odin, each association requires a VAP transfer, where the number of transfers depends on the number of associations. Therefore, both the frequency and the scale of reassociation should be taken into account for various types of architectures;

– *Deployment cost*: it is ideal to define decision metrics that do not require AP or client modification. For example, although BestAP claims low measurement overhead in terms of delay and bandwidth, it requires the modification of APs and clients.

Unfortunately, the existing AsC mechanisms do not take into account the effects of these three factors, the scalability of the proposed mechanisms has not been studied thoroughly, and their effect on the energy consumption of smartphones and resource constraint devices is unknown.

Although some AsC mechanisms propose their own measurement techniques, continuous and low-overhead network monitoring should be provided as a SDWLAN architectural service to facilitate the development and interoperability of control mechanisms. For example, as we will see in Section V, dynamic ChA mechanisms share some of the metrics used by AsC mechanisms. We propose the followings to provide network applications with low-overhead and continuous measurement of metrics: (i) designing metrics that can be used by multiple control mechanisms; (ii) efficient encoding of control data conveyed by a south-bound protocol, (iii) novel hardware/software techniques deployed on network infrastructure devices for passive (i.e., without introducing extra traffic) measurement of network status, and (iv) inference and prediction algorithms that extract new metrics and provide insight into the future status of the network. Compared to distributed implementations, SDWLANs enable in-depth analysis of clients' traffic type and pattern by looking into packet headers [30], [175]–[178]. This information could be employed to improve the accuracy of traffic prediction and interference modeling. Specifically, while the existing approaches rely on the saturated demands of clients when estimating load and interference levels, the throughput and airtime of clients and APs strongly depend on the protocols used at the transport and application layers [172], [179], [180]. For example, the periodic nature of an IoT device's uplink traffic could be exploited to improve AsC performance. We believe that learning and prediction mechanisms must be employed by SDWLANs to provide network control applications (such as AsC and ChA) with meaningful, realistic, and predictive knowledge regarding network operation.

Hybrid design is another approach towards achieving timely reactions against dynamics. The overhead of network monitoring and control can be improved through hybrid control mechanisms, as proposed by SoftRAN [181]. Due to the high dynamics of wireless networks, it is desirable to design control mechanisms that can make control decisions locally, while improving decisions based on the global network view as well. This solution could also benefit from hierarchical controller topologies to make decisions at multiple levels, depending on the dynamicity of variations. For example, AeroFlux (Section III-C) proposes an architecture to enable the design of multi-level control mechanisms. Such mechanisms make control decisions in a distributive manner, at local controllers or by a central controller, based on network dynamics, time constraints, and overhead of exchanging control messages.

*4) **Security Considerations**:* As centralized mechanisms rely on global network information and usually aim to achieve a network-wide optimization, receiving malicious reports has a more severe effect on performance, compared to distributed control. Verifying the legitimacy of network status reports and clients' demands have not yet been addressed by SDWLAN architectures and AsC mechanisms.

V. Centralized Channel Assignment (ChA)

In this section, we review centralized ChA mechanisms. To be consistent with Section IV, we focus on the metrics employed as well as the problem formulation and solution proposed by these mechanisms. In addition, we highlight the performance improvements achieved, compared to distributed mechanisms.

Considering the broadcast nature of wireless communication as well as the dense deployment of APs, ChA is an important network control mechanism to reduce co-channel interference and improve capacity. Basically, a WLAN can be modeled as a conflict graph where the vertices represent APs, and edges reflect the interference level between APs. The color of each vertex is the channel assigned to that AP. The objective of a ChA mechanism is to color the graph using a minimum number of colors in order to improve channel reuse.

A typical channel assignment algorithm relies on all or a subset of the following inputs: (i) the set of APs and clients, (ii) AP-client associations, (iii) interference relationship among APs and clients, and (iv) total number of available

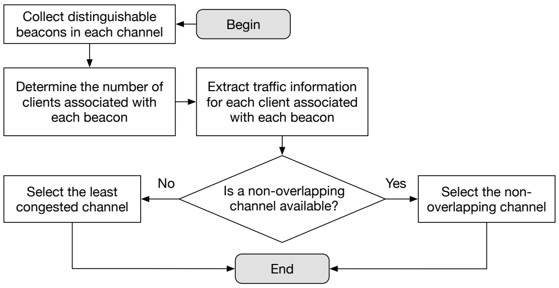
Begin
Collect distinguishable beacons in each channel
Determine the number of clients associated with each beacon
Extract traffic information for each client associated with each beacon
Is a non-overlapping channel available?
No
Yes
Select the least congested channel
Select the non-overlapping channel
End

Fig. 10. Least Congested Channel Search (LCCS) algorithm [182].

channels. Using the aforementioned information, optimization metrics, such as interference, throughput, spectral efficiency, and fairness, are used to define an optimization problem. In this section we study centralized ChA mechanisms.

Figure 10 represents the flowchart of the **least congested channel search (LCCS)** [182] algorithm. Although LCCS is not a central ChA mechanism, we briefly discuss its operation as it has been used by commercial off-the-shelf APs and adopted as the baseline for performance comparison of various ChA mechanisms. In this algorithm, APs scan all channels passively after startup. Each AP collects the beacon frames from its neighboring APs and extracts the number of connected clients and their traffic information. Then, the AP selects a non-overlapping channel if it is available. Otherwise, the AP selects the channel with lowest traffic sum. To obtain client and traffic information, LCCS relies on the use of optional fields in beacon packets. Therefore, the use of this mechanism requires vendor support.

A simpler approach, which has been adopted as the comparison baseline by some ChA mechanisms, is to simply sample the RSSI of nearby APs on all channels and choose the channel with the minimum RSSI level. The literature refers to this approach as **minRSSI**.

We overview ChA mechanisms, categorized into two groups: (i) *traffic-agnostic*, and (ii) *traffic-aware*, as follows.

A. Traffic-Agnostic Channel Assignment

These mechanisms do not include the traffic load of APs and the traffic demand of clients in their channel assignment process. In fact, these mechanisms mainly rely on the interference relationship to formulate their graph coloring problems.

Weighted coloring channel assignment (WCCA). This seminal work [183] formulates channel assignment through the weighted graph coloring problem and improves channel re-use by assigning partially-overlapping channels to APs. Vertices are APs, and edges represent the potential interference between corresponding APs (see Figure 11). The colors of the vertices represent the assigned channel number to APs. The number of clients associated with two neighboring APs and the interference level between them are used to calculate the weight of conflicting edges. *Site-Report* is the proposed method to construct the overlap graph and capture network dynamics. This method enables APs to request their clients perform channel scanning. A list of active APs is generated for each channel. This list includes the APs in direct communication range and the APs whose associated clients are in direct communication with the client performing Site-Report scan. Based on the results of Site-Report, $W(AP_i, AP_j)$ is computed as follows,

$$W(AP_i, AP_j) = \frac{N_{AP_i,AP_j} + N_{AP_j,AP_i}}{N_{AP_i} + N_{AP_j}}, \qquad (10)$$

where N_{AP_j} is the number of site reports performed by the clients of AP_j, and N_{AP_i,AP_j} is the number of site reports of AP_i's clients that reported interference with AP_j or its associated clients.

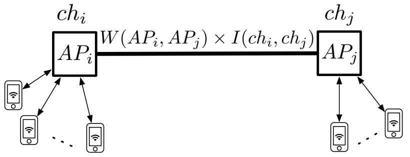

Fig. 11. Weighted coloring channel assignment (WCCA) [183] formulates a weighted graph coloring problem. This figure represents the I-value of the conflicting edge between AP_i and AP_j.

The I-factor, $I(ch_i, ch_j)$, is the normalized interference factor between two channels ch_i and ch_j. The value of $I(ch_i, ch_j)$ is computed as follows: (i) for two non-overlapping channels (e.g., channel 1 and 6 in the 2.4GHz band) the value is 0; (ii) if $ch_i = ch_j$, the value is 1; (iii) for two partially-overlapping channels ch_i and ch_j, the value is between 0 and 1, depending on the distance between their central frequency.

The weight of the conflicting edge between AP_i and AP_j is defined by $Ivalue(AP_i, AP_j) = W(AP_i, AP_j) \times I(ch_i, ch_j)$. Having K colors (channels), the problem is determining how to color the graph with minimum number of colors in order to optimize the objective function. Three different objective functions, O_{max}, O_{sum} and O_{num}, were used. O_{max} represents the maximum I-value amongst all the links. O_{sum} is the sum of I-values of all conflicting edges. The total number of conflicting edges is represented by O_{num}. In order to minimize the objective functions, two distributed heuristic algorithms are proposed: (i) *Hminmax* aims to minimize O_{max} without AP coordination, and (ii) *Hsum* aims to jointly minimize O_{max} and O_{sum}, which requires coordination between APs in order to minimize the interference level of all conflicting edges. Although the proposed algorithms were run distributively, it would be easier to provide them with their required information using a SDWLAN architecture.

The proposed algorithms have been evaluated using NS2 [184] simulator and testbed. Two different scenarios are used in simulations: (i) three non-overlapping channels and (ii) partially overlapping channels (11 channels in the 2.4GHz band). For the first scenario, the algorithms showed a 45.5% and 56% improvement in interference reduction for sparse and dense networks, respectively. For the second scenario, there is a 40% reduction in interference, compared to LCCS.

Client-driven channel assignment (CDCA). [185] formulates a *conflict set coloring* problem and proposes a heuristic algorithm that increases the number of conflict-free clients

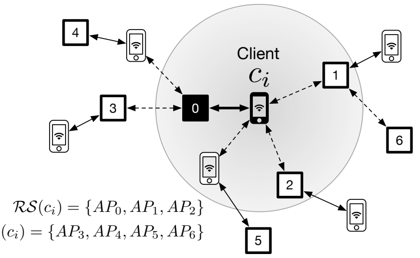
4
Client
c_i
1
3
0
6
2
$\mathcal{RS}(c_i) = \{AP_0, AP_1, AP_2\}$
$\mathcal{IS}(c_i) = \{AP_3, AP_4, AP_5, AP_6\}$
5

Fig. 12. Client-driven channel assignment (CDCA) [185] assigns two sets per client: *range set* (denoted by $\mathcal{RS}(c_i)$) and *interference set* (denoted by $\mathcal{IS}(c_i)$). APs are specified by numbered squares.

at each iteration. Here, *conflict* refers to the condition in which two nodes (APs or clients) that belong to different BSSs use the same channel. To measure the interference level experienced by each client, measurement points are selected, and the signal level received from each AP is measured at those points. Based on this information, there are two sets assigned to each client: (i) *range set*: the APs that can communicate with the client directly, and (ii) *interference set*: the APs that cannot communicate with the client, but can cause interference. Figure 12 shows the range set and interference set for a given client.

This work defines two objective functions: (i) one that maximizes the number of conflict-free clients, and (ii) one that minimizes the total conflict in the network. To achieve the first objective, if a channel ch is assigned to an AP, then no AP in the range set and interference set of the clients associated with that AP should have channel ch assigned to them. A client satisfying this condition is referred to as a ”conflict-free client”. Since it may not be possible to satisfy this requirement for all the clients, the algorithm minimizes the overall network conflict.

The main step of the channel assignment algorithm randomly selects an AP and assigns a channel that results in the maximum number of conflict-free clients. This step is repeated for all APs, and the order of APs is determined by a random permutation. This process is repeated as long as it increases the number of conflict-free clients.

After the first step, there might be clients that are left with a high level of interference. To remedy this condition, the conflict level of all non-conflict-free clients is balanced. This mechanism implicitly results in traffic load balancing.

Frequency planning in large-scale networks (FPLN). [28] models the channel assignment problem as a weighted graph coloring problem taking into account the external interference generated by non-controllable APs. The interference level caused by AP_j on AP_i, which is the weight of edge $AP_j \to AP_i$, is defined as follows,

$$W(AP_i, AP_j) = \mathcal{C}_{AP_i}^{active} \times I(ch_i, ch_j) \times P(AP_i, AP_j), \quad (11)$$

where $\mathcal{C}_{AP_i}^{active}$ is the number of active associated clients with AP_i. An associated client is called *active* in a time interval if it is sending/receiving data packets. The interference factor between two channels assigned to AP_i and AP_j is given by $I(ch_i, ch_j)$. The power received at AP_j from AP_i is represented by $P(AP_i, AP_j)$. The interference factor of AP_i is defined as $\sum_{\forall AP_j \in \mathcal{AP}} W(AP_i, AP_j)$. The objective function is defined as the sum of all APs' interference factor, including controllable and non-controllable APs. The ChA mechanism aims to minimize the objective function when the channels of non-controllable APs are fixed.

The authors propose a two-phase heuristic algorithm to solve the above optimization problem. In the first phase, the controllable APs are categorized into separate clusters based on neighborhood relationships, and the local optimization problem is solved for each cluster by finding an appropriate channel per AP to minimize its interference level while the channels of other APs (in the cluster) are fixed. This process is started from the AP with largest interference factor. In the second phase, a pruning-based exhaustive search is run on each cluster, starting from the AP with highest interference level. The algorithm updates the channels of APs to reduce the total interference level of the cluster. The pruning strategy deletes the failed channel assignments (i.e., the solutions that increase the interference level obtained in the first phase) from the search space. FPLN uses CAPWAP (see Section III-C1) as its south-bound protocol. Simulation and testbed experiments show about a 1.2x lower interference level, compared to LCCS.

CloudMAC. The CloudMAC [26], [50] architecture allows multiple NICs of a physical AP to be mapped to a VAP, where NICs may operate on different channels. These NICs periodically monitor and report channel utilization to the controller. Unfortunately, it is not clear how channel measurement is performed. If a client is operating on a high-interference channel, the controller sends a 802.11h channel switch announcement message to the client (mandatory for 802.11a/n standards), instructing it to switch to another channel with lower interference. Since a client is associated with a VAP, the client does not need to re-associate and it can continue its communication through the same VAP using another NIC of the same physical AP. Additionally, this process does not require any client modification. Unfortunately, the performance evaluation of this mechanism has not been reported.

Odin. [102] proposes a simple channel assignment strategy. For each AP, the channel assignment application (running on Odin controller) samples the RSSI value of all channels during various operational hours. The heuristic algorithm chooses the channel with the smallest maximum and average RSSI for each AP. Unfortunately, the performance of this mechanism has not been evaluated.

Primary channel allocation in 802.11ac networks (PCA). The *hidden channel* (HC) problem is addressed in [29] for 802.11ac networks. The HC problem occurs when an AP interferes with the bandwidth of another AP. This problem occurs due to the heterogeneity of channel bandwidths, different CCA thresholds for primary and secondary channels, and bandwidth-ignorant fixed transmission power.

Figure 13 illustrates an example of the HC problem. The bandwidth of AP_1 is 80MHz, including four 20-MHz channels 1, 2, 3 and 4, where channel 1 is its primary channel. The primary channel sensing range is greater than the secondary channel sensing range due to the difference in CCA thresholds for primary and secondary channels in 802.11ac. AP_2 uses

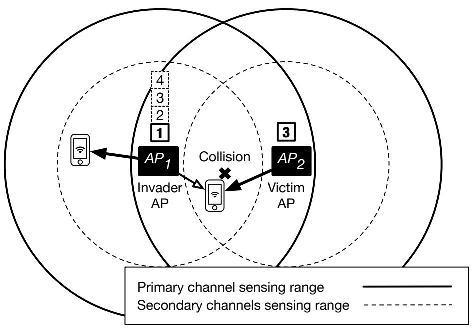
4
3
2
1
3
AP_1
Collision
AP_2
Invader AP
Victim AP
Primary channel sensing range
Secondary channels sensing range

Fig. 13. An example of the hidden-channel (HC) problem in 802.11ac [29].

a 20MHz bandwidth operating on channel 3. Suppose AP_2 is communicating with its clients on channel 3. When AP_1 performs carrier sensing on its channels, it cannot detect the presence of AP_2 on channel 3 because AP_2 is not in the secondary sensing range of AP_1. Therefore, AP_1 starts to send data on channels 1, 2, 3 and 4, and invades AP_2.

In order to study the impact of the HC problem, the authors proposed a Markov chain-based analysis to show the impact of packet length and MAC contention parameters on the performance of APs. Furthermore, the effect of various HC scenarios on packet error rate (PER) has been investigated. The authors used the graph coloring problem, where APs are vertices and channels are colors, to model the *primary channel allocation* (PCA) problem. The edges represent the invasion relationship between APs. It is assumed that a controller collects information about the bandwidth of APs and their interference relationship. The objective is to color the vertices in order to: (i) minimize the interference/invasion relationship between APs, and (ii) maximize channel utilization. The problem is formulated as an integer programming optimization problem, which is NP-hard. A heuristic primary channel allocation algorithm is introduced to solve the problem. Simulation results show the higher performance of PCA compared to minRSSI and random channel assignment. Furthermore, PCA is a close-to-optimal solution when compared to the optimal exhaustive search algorithm.

EmPOWER2. As mentioned in Section III-C3, EmPOWER2 [24] establishes *channel quality and interference map* abstraction at the controller. The proposed ChA mechanism uses the APIs provided to traverse the map and detect uplink/downlink conflicts between LVAP pairs. There is an edge between two nodes in the interference map if they are in the communication range of each other. In addition, the weight of each link corresponds to the channel quality between nodes. The graph coloring algorithm proposed in [186] is used to assign channels based on the conflict relationships.

Wi-5 channel assignment (Wi5CA). [187] proposes a ChA mechanism as part of the Wi-5 project [188]. A binary integer linear programming problem is formulated with objective function $\mathbf{U} = \mathbf{G} \times \mathbf{A}^T.\mathbf{I}$, where $'\times'$ and $'.'$ are matrix multiplication and element-wise multiplication operators, respectively. $\mathbf{G} \in \{0,1\}^{|\mathcal{AP}|\times|\mathcal{AP}|}$ represents the network topology. If the average power strength of AP_i on AP_j exceeds a specific threshold, the value of G_{AP_i,AP_j} will be 1, otherwise it will be 0. $\mathbf{A} \in \{0,1\}^{|\mathcal{CH}|\times|\mathcal{AP}|}$ is the current channel assignment for APs. If channel ch_i is assigned to AP_j, the value of A_{ch_i,AP_j} is 1, otherwise it is 0. $\mathbf{I} \in \mathbb{R}^{|\mathcal{AP}|\times|\mathcal{CH}|}$ represents the predicted interference matrix, where I_{AP_i,ch_i} is the interference level predicted for AP_i if channel ch_j is assigned to it. Unfortunately, the interference prediction method is unclear. The channel assignment algorithm minimizes

$$\sum_{\forall AP_i \in \mathcal{AP}} \sum_{\forall ch_j \in \mathcal{CH}} U_{AP_i,ch_j}, \qquad (12)$$

which is the network-wide interference level. A controller executes the ChA mechanism when the interference level is higher than a threshold. MATLAB simulations show 2dB and 3dB reduction in the average interference level, compared to LCCS and uncoordinated channel assignment, respectively.

B. Traffic-Aware Channel Assignment

Traffic-aware channel assignment mechanisms include the traffic of APs, clients or both in their decision process. As measuring the effect of interference on performance is complicated, these mechanisms explicitly include metrics such as throughput, delay, and fairness, in their problem formulations.

AP placement and channel assignment (APCA). [189] addresses AP placement and channel assignment in order to improve the total network throughput and the fairness established among clients. The throughput of clients is estimated as a function of MAC-layer timing parameters, network topology, and data rate of clients. The data rate of each client (11, 5.5, 2 or 1Mbps) depends on the link's RSSI. The fairness among clients is defined as the throughput deviations of clients based on the Jain's fairness index [190] as follows,

$$J = \frac{(\sum_{\forall c_i \in \mathcal{C}} Th_{c_i})^2}{|\mathcal{C}| \sum_{\forall c_i \in \mathcal{C}} (Th_{c_i})^2}, \qquad (13)$$

where Th_{c_i} is the throughput of client c_i. Maximizing the objective function, defined as $J \times \sum_{\forall c_i \in \mathcal{C}} Th_{c_i}$, leads to optimizing both total throughput and fairness among clients. The authors propose a heuristic local search method, named the *patching* algorithm. In each iteration of this algorithm, an AP is placed on one of the predefined locations, and a non-overlapping channel is assigned to the AP to maximize the objective function. This process is repeated until a predefined number of APs are placed. Simulation and testbed results show that the proposed patching algorithm provides close-to-optimal solutions in terms of total throughput and fairness. APCA is used only for the initial network configuration phase.

Traffic-aware channel assignment (TACA). [191] assigns weights to APs and clients proportional to their traffic demands. The *weighted channel separation* (WCS) metric is

$$WCS = \sum_{\substack{i,j \in \mathcal{AP} \cup \mathcal{C}, \\ BSS(i) \neq BSS(j)}} W(i,j) \times Separation(i,j), \qquad (14)$$

where $BSS(k)$ is the BSS including AP or client k, $Separation(i,j)$ is the distance between operating channels of node i and node j, and $W(i,j)$ is a function of the traffic demands of i and j. $Separation(i,j) = \min(|ch_i - ch_j|, 5)$, therefore, the maximum separation value is 5, which is the

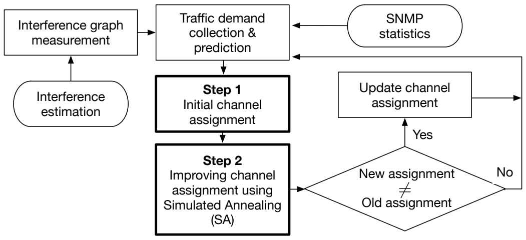
Interference graph measurement
Traffic demand collection & prediction
SNMP statistics
Interference estimation
Step 1
Initial channel assignment
Update channel assignment
Yes
Step 2
Improving channel assignment using Simulated Annealing (SA)
New assignment ≠ Old assignment
No

Fig. 14. Traffic-aware channel assignment (TACA) [191].

distance between orthogonal channels (e.g., channel 1 and 6). Since all nodes are included in the calculation of the WCS metric, all AP-AP, AP-client and client-client interferences are taken into account.

A heuristic algorithm has been proposed to maximize WCS. Figure 14 shows the flowchart of this algorithm. The interference graph measurement is performed a few times per day. The traffic demands of APs and clients are collected and predicted using the SNMP statistics collected from APs. In ”Step 1”, an initial channel assignment is performed to maximize the WCS metric considering APs’ traffic demands only. In ”Step 2”, a simulated annealing (SA) approach is used to update channel assignment, which maximizes WCS by taking into account the traffic demands of AP and clients. This step randomly selects an AP and its clients, and assigns a new channel at each iteration in order to maximize the WCS metric. Empirical results (using a 25-node testbed) show 2.6 times increase in TCP throughput, compared to a downgraded algorithm in which traffic demands are not included. The authors also showed that one channel switching per 5 minutes does not result in throughput degradation.

Virtual delayed time channel assignment (VDTCA). [27] introduces a new interference prediction model based on the signal strength and traffic demands of APs and clients. The extra transmission delay caused by interference between two APs is represented as $VDT = T_{int} - T_n$, where VDT is *virtual delayed time*. The term ”virtual” refers to the calculation of this metric through interference prediction without actually changing channels. T_{int} and T_n are the time required to transmit a given amount of traffic in the presence and absence of interference, respectively.

The transmission time is a function of physical layer data rate, which depends on the signal SINR computed through measuring RSSI at receiver side. On the other hand, the traffic demands of APs and clients have a great impact on the transmission time of neighboring APs. For instance, when there is no traffic on an AP, there will be no transmission delay overhead on the neighboring APs and clients. Therefore, T_{int} is calculated based on RSSI values and traffic rates of all neighboring APs and their associated clients. Using the proposed VDT model, it is possible to measure the VDT of each AP pair. Channel assignment is modeled as a graph vertex coloring problem, which uses $VDT(BSS_i, BSS_j)$ as the edge weight between two interfering BSSs, where BSS_i and BSS_j are two APs and their associated clients. Non-overlapping channels are the colors of vertices. The objective is to color the graph with a minimum number of colors while minimizing the sum of edge weights to achieve interference minimization. The semi-definite programming (SDP) [192] relaxation technique is used to solve this problem. The authors show that SDP can solve the channel assignment problem in the order of seconds for a medium-size network. For instance, it takes 27.5 seconds to find the solution for a network with 50 APs and 9 non-overlapping channels. Testbed results show 30% throughput improvement, compared to LCCS [182].

Cisco unified wireless network (CUWN). In the CUWN architecture [85] (see Section III-C1), based on the different RF groups established and the cost metric calculated for each AP, each controller updates the assigned channels periodically[4]. The controller calculates a cost metric for each AP to represent the interference level of the AP, and prepares a list called *channel plan change initiator* (CPCI), which includes the sorted cost metric values of all APs. The leader selects the AP with highest cost metric and assigns a channel to the selected AP and its one-hop neighboring APs in order to decrease their cost metrics. These APs are removed from the CPCI list, and this process is repeated for all remaining APs in the list. A sub-optimal channel assignment is calculated through a heuristic algorithm[5] that aims to maximize the frequency distance of selected channels. Note that a longer frequency distance results in a lower interference between APs. CUWN also enables network administrator to set channels manually.

Channel assignment with fairness approach (CAFA). [193] formulates ChA as a weighted graph coloring problem where weights are assigned to the set of edges and vertices. The distance of overlapping channels is determined by the normalized interference factor, i.e., I-factor, introduced in [183] (see WCCA explained in Section V-A). The normalized throughput of each client is estimated using the approximation introduced in [194]. The weight of AP_i is defined as the total *normalized throughput reduction* of AP_i, which is calculated based on clients’ normalized throughput and I-factor. The weight of the edge between two interfering APs represents the throughput reduction caused by their mutual interference. Furthermore, the authors formulate the fairness among normalized throughputs of APs using Jain’s fairness index [190]. The objective function is defined as the joint minimization of clients’ throughput reduction and maximization of client’s fairness index.

An iterative heuristic algorithm is introduced to solve the optimization problem. The algorithm sorts the APs based on their weights (i.e., normalized throughput reduction). At each iteration, the AP with highest weight is selected for channel assignment. The channel of the selected AP is assigned so that the throughput reduction of the neighboring APs is minimized. This process is repeated until channel assignment is performed for all APs. Simulation results show 15% and 6% improvement in total throughput for 4-AP and 8-AP scenarios, respectively, compared to WCCA [183].

Throughput and fairness-aware channel assignment

[4] The period can be adjusted by network administrators.

[5] The details of ChA mechanism used by CUWN are not available.

(TFACA). [195] uses spectrum monitoring information and the traffic demand of APs. This mechanism defines the utility function of AP_i when operating on channel ch_j as,

$$U(AP_i, ch_j) = \frac{\min\{F(AP_i, ch_j), T(AP_i)\}}{T(AP_i)}, \tag{15}$$

where $F(AP_i, ch_j)$ is the free airtime of channel ch_j on AP_i, and $T(AP_i)$ is the total time required to send data of AP_i in a time unit. Note that Equation 15 represents the percentage of AP_i's data that can be sent over channel ch_j in a given time unit. Two objective functions are defined:

$$\max \sum_{\forall AP_i \in \mathcal{AP}, \forall ch_j \in \mathcal{CH}} U(AP_i, ch_j), \tag{16}$$

and,

$$\max(\min_{\forall AP_i \in \mathcal{AP}, \forall ch_j \in \mathcal{CH}} U(AP_i, ch_j)). \tag{17}$$

A heuristic algorithm has been proposed to solve the above NP-hard problem in the following two steps: (i) temporary channel assignment, which assigns a channel to each AP independent of other APs, to maximize each AP's utility function, and (ii) channel reassignment of the APs with low utility function (starting with the lowest one) without decreasing the sum of all utility function values. The second step is repeated for a given number of iterations or until $U(AP_i, ch_j) = 1$ for all APs. A fairness index similar to Equation 13 has been defined to include the utility function of APs.

The 802.11g protocol with non-overlapping channels are used for performance evaluation through simulation and testbed. An 8 to 15$\%$ improvement in the fairness index and an 8 to 21$\%$ increase in the mean utility function are achieved, compared to minRSSI.

Frequency and bandwidth assignment for 802.11n/ac (FBWA). [196] addresses channel and bandwidth allocation to benefit from the channel bonding feature of 802.11n/ac. The utility function of AP_i that works on primary channel ch_j and bandwidth β is defined as

$$U(AP_i, ch_j, \beta) = \frac{Th(AP_i, ch_j, \beta)}{\min\left\{Th^{\max}(AP_i, ch_j, \beta),\ L^{\text{mean}}_{\text{AP}_i}\right\}} \tag{18}$$

where $Th(AP_i, ch_j, \beta)$ is the expected throughput of AP_i operating on primary channel ch_j and channel bandwidth β, $Th^{\max}(AP_i, ch_j, \beta)$ is the maximum achievable throughput of AP_i, and $L^{\text{mean}}_{\text{AP}_i}$ is the mean traffic load on AP_i. Through a heuristic algorithm similar to TFACA [195], the appropriate primary channels and bandwidths are determined for all APs to maximize the sum of the APs' utility functions.

Testbed results show a 65 to 89$\%$ reduction in the number of overlapping BSSs, and a mean throughput that is 3 times higher, compared to random channel assignment and minRSSI.

Normalized-airtime channel assignment (NATCA). [197] requires each AP to measure its busy time (referred to as *normalized airtime*) and report to a controller periodically. The busy time of an AP is the percentage of time that a channel assigned to that AP is occupied by its neighbors. An AP measures this value when sending data towards its clients. A threshold value is defined to categorize APs into two groups based on their busy time: *heavily congested* and *lightly congested*. An optimization problem has been proposed to minimize the sum of the normalized airtime of all APs without modifying the status of lightly congested APs. A Tabu search algorithm is proposed to solve the optimization problem. NS3 simulations show a 1.5x throughput improvement, compared to LCCS [182]. The measurement of normalized airtime for all APs is performed passively using the NS3 tracing system, which provides packet-level trace files for all APs.

C. Channel Assignment: Learned Lessons, Comparison, and Open Problems

Table IV presents and compares the features of ChA mechanisms. In the following, we study these features and identify research directions.

*1) **Dynamicity and Traffic-Awareness**:* As Table IV shows, not all the ChA mechanisms support dynamic channel reallocation. In addition, most of the ChA mechanisms are either traffic-agnostic, or they do not recognize uplink traffic. Furthermore, many of the ChA mechanisms do not actually discuss the data gathering policy employed, and the rest focus on the mean traffic load of APs and the mean throughput demand of clients. These limitations are due to two main reasons: First, the higher the rate of central interference map generation, the higher the overhead of control data exchanged by the controller. Second, although heuristic algorithms have been proposed to tackle the NP-hardness of the channel assignment problems, the execution duration of these algorithms might not be short enough to respond to network dynamics. The two approaches earlier proposed in Section IV-C3 (i.e., prediction and hybrid design) can be employed to cope with these challenges. For example, depending on the time window and accuracy of predictions, a ChA proactively responds to dynamics, which in turn reduces the burden on real-time network mapping and fast algorithm execution. The second solution, i.e., hybrid design, employs multi-level decision making to support fast and low-overhead reaction to network dynamics. For the hybrid designs, however, the topology of controllers may be tailored depending on the control mechanism employed. For example, while a suitable controller topology for a ChA mechanism depends on the interference relationship between APs, a more suitable topology for an AsC mechanism is to connect all the APs of a hallway to a local controller to improve the QoS of clients walking in that area. Therefore, it is important to design architectures that support flexible communication between controllers.

*2) **Joint ChA and AsC**:* As channel switching may lead to re-association (depending on the architecture used), clients may experience communication interruption and violation of their QoS requirements [111], [198], [199]. Therefore, it is necessary to measure and reduce the channel switching overhead of dynamic ChA mechanisms. In this regard, joint design of ChA and AsC is essential for uninterrupted performance guarantee. Such a joint mechanism, for example, monitors and predicts clients' traffic pattern, performs channel assignment to maximize spatial reuse, and moves VAPs between APs to support seamless handoff as the traffic changes. In addition, mobility prediction [200], [201] may be employed to minimize

TABLE IV
COMPARISON OF CHANNEL ASSIGNMENT (CHA) MECHANISMS

Mechanism	Traffic-Aware		Standards & Channels		**Dynamic**	Performance Evaluation	
	Downlink	**Uplink**	**Standard**	**Partially Overlapping**		**Simulation**	**Testbed**
WCCA [183]	×	×	802.11b	✓	✓	✓	✓
CDCA [185]	×	×	802.11b	×	✓	✓	✓
FPLN [28]	×	×	802.11b	✓	×	✓	✓
CloudMAC [26]	×	×	802.11a/b/g/n	✓	✓	×	✓
Odin [102]	×	×	802.11a/b/g/n	✓	✓	×	✓
PCA [29]	×	×	802.11ac	×	✓	✓	×
EMPOWER2 [24]	×	×	802.11a/b/g/n/ac	×	✓	×	✓
Wi5CA [187]	×	×	802.11b/g	×	✓	✓	×
APCA [189]	×	✓	802.11b	×	×	✓	×
TACA [191]	✓	✓	802.11b/g	✓	✓	✓	✓
VDTCA [27]	✓	✓	802.11a/g	×	✓	×	✓
CUWN [85]	✓	✓	802.11a/b/g/n/ac	✓	✓	×	×
CAFA [193]	✓	✓	802.11b	✓	×	✓	×
TFACA [195]	✓	×	802.11g	×	×	✓	✓
FBWA [196]	✓	×	802.11a/n/ac	×	×	✓	×
NATCA [197]	✓	×	802.11g	×	✓	✓	×

the delay of AsC and ChA by enabling the use of proactive control mechanisms instead of using reactive mechanisms that are triggered by performance drop. Note that mobility prediction is specifically useful in SDWLANs as these architectures enable the central collection and analysis of metrics such as RSSI to apply localization and mobility prediction algorithms. Unfortunately, the existing joint ChA-AsC mechanisms (e.g., [202], [203]) operate distributively and do not provide any performance guarantee, which is required for mission-critical applications such as medical monitoring or industrial control [17].

3) ***Partially Overlapping Channels and AP Density Management****:* Most ChA mechanisms designed for 2.4GHz networks utilize only non-overlapping channels to decrease the interference level (as shown in Table IV). However, due to the dense deployment of APs, this approach limits the tradeoff between interference reduction and channel reuse [204]. To address this concern, research studies have improved the performance of channel assignment by dynamically assigning partially-overlapping channels [193]. These mechanisms, for example, rely on the interference relationship between APs to increase the distance between assigned channels as the pairwise interference increases. However, these mechanisms require the underlying SDWLAN to provide APIs for efficient collection of network statistics.

In addition to utilizing partially-overlapping channels, ChA mechanisms can further improve network capacity through AP topology management, which is achieved by three main strategies: (i) *AP placement*: the site survey is performed to determine the location of APs during the installation phase [205]; (ii) *AP power control*: a SDWLAN controller dynamically adjusts the transmission power level of APs [206]; (iii) *AP mode control*: a SDWLAN controller dynamically turns on/off APs [107]. Note that the second and third strategy require architectural support. It is also important to coordinate the aforementioned strategies with AsC to avoid clients' service interruption. For example, by relying on the global network view established, clients association may be changed before channel assignment or transmission power control cause client disconnection. Although these problems have been addressed in isolation or distributively [40], [207]–[209], centralized and integrated solutions are missing.

4) ***Integration with Virtualization****:* Our review of ChA and AsC mechanisms shows that these mechanisms are oblivious to virtualization. Specifically, they do not take into account how the pool of resources is assigned to various network slices. However, virtualization has serious implications on network control. For example, mobility management becomes more challenging when network virtualization is employed. When a client requires a new point of association due to its mobility, in addition to parameters such as fair bandwidth allocation, the available resources of APs should also be taken into account. More specifically, for a client belonging to slice n, the AsC mechanism should ensure that after the association of this client with a new AP, the QoS provided by slice n and other slices is not violated. However, as this may require client steering, the re-association cost of other clients should be minimized. As an another example, a ChA mechanism may assign non-overlapping and overlapping channels to a mission-critical slice and a regular slice, respectively. The problem becomes even more complicated when multiple controllers collaborate to manage network resources. The integration of network slicing and control mechanisms would pave the way to use SDWLANs in emerging applications, such as mission-critical data transfer from mobile and IoT devices.

In addition to network-based slicing of resources, it is desirable to support personalized mobility services as well. To this end, AsC mechanisms require architectural support to enable client association based on a variety of factors. As an example, a client's VAP stores the client's handset features and exposes that information to the AsC mechanism to associate the client with APs that result in minimum energy consumption of the client. Such interactions between architectural components and control mechanisms are unexplored.

Another important challenge of slicing at low level is the complexity of coordination. For example, assume that different transmission powers are assigned to flows. In this case, simply changing the transmission power of APs working on the same channel would cause significant interference and disrupt the services being offered. Therefore, offering low-level virtualization requires coordination between the network control mechanisms.

*5) **Coexistence of High-Throughput and Legacy Clients**:* Although new WiFi standards (e.g., 802.11n/ac) supporting high throughput are broadly used in various environments, we should not neglect the existence of legacy devices (e.g., 802.11b), especially for applications that require low throughput [1], [37], [38]. In heterogeneous networks, the throughput of high-rate 802.11ac devices is jeopardized by legacy clients, due to the wider channel widths used by 802.11n/ac through channel bonding [210], [211]. Specifically, the channel bonding feature of 802.11n/ac brings about new challenges such as *hidden-channel problem*, *partial channel blocking*, and *middle-channel starvation* [29], [211], [212]. The bandwidth of a 802.11n/ac device (80/160MHz) is blocked or interrupted by a legacy client (20MHz) due to the partial channel blocking and hidden-channel problem. In the middle-channel starvation problem, a wideband client is starved because its bandwidth overlaps with the channels of two legacy clients. Hence, the wideband client can use its entire bandwidth only if both legacy clients are idle. Despite the heterogeneity of WLAN environments, our review shows that addressing the challenges of multi-rate networks to satisfy the diverse set of QoS requirements is still premature. In fact, only PCA [29] proposes a central solution, and its performance has been compared with distributed mechanisms.

*6) **Security**:* Wireless communication is susceptible to jamming attacks through which malicious signals generated on a frequency band avoid the reception of user traffic on that channel. The vulnerability of 802.11 networks to such denial of service (DoS) attacks have been investigated through various empirical measurements [213]–[215]. Although the research community proposes various mechanisms (such as frequency hopping [216]) to cope with this problem, the existing approaches do not benefit from a central network view. In particular, by relying on the capabilities of SDWLANs, clients and APs can periodically report their operating channel information (such as channel access time, noise level and packet success rate) to the controller, and mechanisms are required to exploit this information to perform anomaly detection.

VI. Related Surveys

In this section we mention the existing surveys relevant to SDN and WLAN.

A comprehensive study of software-defined wireless cellular, sensor, mesh, and home networks is presented in [217]. This work classifies the existing contributions based on their objective and architecture. A survey of data plane and control plane programmability approaches is given in [218], where the authors review the components of programmable wired and wireless networks, as well as control plane architectures and virtualization tools. [219] presents a survey of virtualization and its challenges in wireless networks. A survey of OpenFlow concepts, abstractions and its potential applications in wired and wireless networks is given in [220]. [3] and [221] review the 802.11 standards for high-throughput applications, TV white spaces and machine-to-machine communications. [39] surveys energy-efficient MAC protocols for WLANs. A survey of cellular to WLAN offloading mechanisms is given in [4].

Channel assignment strategies are reviewed in [222], and the dynamics of channel assignment in dense WLANS are investigated in [223]. QoS support in dense WLANs through layer-1 and layer-2 protocols are surveyed in [224]–[226]. Client-based association mechanisms and AP placement strategies are surveyed in [227]. Distributed load-balancing mechanisms are reviewed in [228]. [229] provides a survey on spectrum occupancy measurement techniques in order to generate interference maps in WLANs. In [230], the future demands of WLANs and the potential new features of ongoing 802.11ax standard are investigated and reviewed. A review of wireless virtualization mechanisms is given in [231]. This work also presents the business models, enabling technologies, and performance metrics of wireless virtual networks.

Our review of existing surveys shows that there is no study of WLANs from a SDN point of view. Therefore, the main motivation for conducting this study was the lack of a comprehensive survey on SDWLAN architectures and essential centralized network control mechanisms.

VII. Conclusion

SDWLANs enable the implementation of network control mechanisms as applications running on a network operating system. Specifically, SDWLAN architectures provide a set of APIs for network monitoring and dissemination of control commands. Additionally, network applications can benefit from the global network view established in the controller to run network control mechanisms in a more efficient way, compared to distributed mechanisms. In this paper, we categorized and reviewed the existing SDWLAN architectures based on their main contribution, including *observability and configurability*, *programmability*, *virtualization*, *scalability*, *traffic shaping*, and *home networks*. Through comparing the existing architectures as well as identifying the growing and potential applications of SDWLANs, we proposed several research directions, such as establishing trade-off between centralization and scalability, investigating and reducing the overhead of south-bound protocols, exploiting layer-1 and layer-2 programmability for virtualization, and opportunities and challenge of SDWLANs with respect to security provisioning.

After reviewing architectures, we focused on association control (AsC) and channel assignment (ChA), which are the two widely-employed network control mechanisms proposed to benefit from the features of SDWLAN architectures. We reviewed centralized AsC and ChA mechanisms in terms of their metrics used, problem formulation and solving techniques, and the results achieved compared to distributed mechanisms.

At a high level, we categorized AsC mechanisms into two groups: *seamless handoff*, used to reduce the overhead of client handoff, and *client steering*, to improve the performance of clients. Furthermore, we specified several research directions towards improving AsC mechanisms, such as the design and measurement of metrics to include both uplink and downlink traffic into the decision making process, establishing proportional demand-aware fairness among clients, and including the application-level demand and behavior of clients in formulating AsC problems.

We reviewed central ChA mechanisms, and we focused on the approaches employed by these mechanisms to model the interference relationship among APs and clients. Based on the inclusion of traffic in the decision making process, we categorized ChA mechanisms into *traffic-aware* and *traffic-agnostic*. Our comparisons revealed that, given the heterogeneity of clients, the new ChA mechanisms should address the coexistence of low- and high-throughput devices. We also discussed the challenges and potential solutions for the integration of virtualization with AsC and ChA, as well as the importance and potential benefits of AsC and ChA co-design.

VIII. Acknowledgment

This research is partially supported by a grant from the Cypress Semiconductor Corporation (Grant No. CYP001).

References

[1] S. Tozlu, M. Senel, W. Mao, and A. Keshavarzian, “Wi-Fi enabled sensors for internet of things: A practical approach,” *IEEE Communications Magazine*, vol. 50, no. 6, pp. 134–143, 2012.

[2] M. Baird, B. Ng, and W. Seah, “WiFi Network Access Control for IoT Connectivity with Software Defined Networking,” in *8th ACM on Multimedia Systems Conference (MMSys)*, 2017, pp. 343–348.

[3] H. A. Omar, K. Abboud, N. Cheng, K. R. Malekshan, A. T. Gamage, and W. Zhuang, “A survey on high efficiency wireless local area networks: Next generation wifi,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2315–2344, 2016.

[4] Y. He, M. Chen, B. Ge, and M. Guizani, “On wifi offloading in heterogeneous networks: Various incentives and trade-off strategies,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2345–2385, 2016.

[5] “Wi-fi alliance publishes 7 for ’17 wi-fi predictions.” [Online]. Available: http://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-publishes-7-for-17-wi-fi-predictions

[6] “Ericsson mobility report,” 2016. [Online]. Available: https://www.ericsson.com/res/docs/2016/ericsson-mobility-report-2016.pdf

[7] “Global wi-fi enabled equipment shipments from 2012 to 2017 (in billion units).” [Online]. Available: https://www.statista.com/statistics/282512/global-wi-fi-enabled-equipment-shipments/

[8] “Cisco visual networking index: Global mobile data traffic forecast update, 2015-2020 white paper.” [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html

[9] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, “Mobile data offloading: How much can wifi deliver?” *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, pp. 536–550, 2013.

[10] P. Serrano, P. Salvador, V. Mancuso, and Y. Grunenberger, “Experimenting with commodity 802.11 hardware: Overview and future directions,” *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 671–699, 2015.

[11] S. Biswas, J. Bicket, E. Wong, R. Musaloiu-E, A. Bhartia, and D. Aguayo, “Large-scale measurements of wireless network behavior,” in *SIGCOMM*, 2015, pp. 153–165.

[12] R. Murty, J. Padhye, R. Chandra, A. Wolman, and B. Zill, “Designing high performance enterprise wi-fi networks,” in *5th USENIX Symposium on Networked Systems Design and Implementation*, April 2008, pp. 73–88.

[13] I. Broustis, K. Papagiannaki, S. V. Krishnamurthy, M. Faloutsos, and V. P. Mhatre, “Measurement-driven guidelines for 802.11 wlan design,” *IEEE/ACM Transactions on Networking*, vol. 18, no. 3, pp. 722–735, 2010.

[14] K. Sui, S. Sun, Y. Azzabi, X. Zhang, Y. Zhao, J. Wang, Z. Li, and D. Pei, “Understanding the impact of ap density on wifi performance through real-world deployment,” in *IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, 2016.

[15] J. Choi and K. G. Shin, “Qos provisioning for large-scale multi-ap wlans,” *Ad Hoc Networks*, vol. 10, no. 2, pp. 174–185, 2012.

[16] F. Tramarin, S. Vitturi, M. Luvisotto, and A. Zanella, “On the use of ieee 802.11 n for industrial communications,” *IEEE Transactions on Industrial Informatics*, vol. 12, no. 5, pp. 1877–1886, 2016.

[17] B. Dezfouli, M. Radi, and O. Chipara, “REWIMO: A Real-Time and Low-Power Mobile Wireless Network,” *ACM Transactions on Sensor Networks (TOSN)*, vol. 13, no. 3, 2017.

[18] Y. Jarraya, T. Madi, and M. Debbabi, “A survey and a layered taxonomy of software-defined networking,” *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1955–1980, 2014.

[19] D. B. Rawat and S. R. Reddy, “Software defined networking architecture, security and energy efficiency: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 325–346, 2017.

[20] L. E. Li, Z. M. Mao, and J. Rexford, “Toward software-defined cellular networks,” in *EWSDN*, 2012, pp. 7–12.

[21] H. Ali-Ahmad, C. Cicconetti, A. De la Oliva, M. Draxler, R. Gupta, V. Mancuso, L. Roullet, and V. Sciancalepore, “Crowd: an sdn approach for densenets,” in *European Workshop on Software Defined Networks (EWSDN)*, 2013, pp. 25–31.

[22] T. Chen, M. Matinmikko, X. Chen, X. Zhou, and P. Ahokangas, “Software defined mobile networks: concept, survey, and research directions,” *IEEE Communications Magazine*, vol. 53, no. 11, pp. 126–133, 2015.

[23] I. T. Haque and N. Abu-ghazaleh, “Wireless software defined networking: A survey and taxonomy,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2713–2737, 2016.

[24] R. Riggio, M. K. Marina, J. Schulz-Zander, S. Kuklinski, and T. Rasheed, “Programming abstractions for software-defined wireless networks,” *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 146–162, 2015.

[25] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid, and A. Feldmann, “Opensdwn: Programmatic control over home and enterprise wifi,” in *SIGCOMM Symposium on Software Defined Networking Research*. ACM, 2015, p. 16.

[26] J. Vestin, P. Dely, A. Kassler, N. Bayer, H. Einsiedler, and C. Peylo, “Cloudmac: Towards software defined wlans,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 16, no. 4, pp. 42–45, February 2013.

[27] Y. Liu, W. Wu, B. Wang, T. He, S. Yi, and Y. Xia, “Measurement-based channel management in wlans,” in *IEEE Wireless Communication and Networking Conference (WCNC)*, April 2010, pp. 1–6.

[28] M. Bernaschi, F. Cacace, A. Davoli, D. Guerri, M. Latini, and L. Vollero, “A capwap-based solution for frequency planning in large scale networks of wifi hot-spots,” *Computer Communications*, vol. 34, no. 11, pp. 1283–1293, 2011.

[29] S. Jang and S. Bahk, “A channel allocation algorithm for reducing the channel sensing/reserving asymmetry in 802.11ac networks,” *IEEE Transactions on Mobile Computing*, vol. 14, no. 3, pp. 458–472, March 2015.

[30] L. Oliveira, K. Obraczka, and A. Rodrguez, “Characterizing User Activity in WiFi Networks: University Campus and Urban Area Case Studies,” in *MSWiM*, 2016, pp. 190–194.

[31] A. Patro and S. Banerjee, “Coap: A software-defined approach for home wlan management through an open api,” in *9th ACM workshop on Mobility in the evolving internet architecture (MobiArch ’14)*, September 2014, pp. 31–36.

[32] S. Zehl, A. Zubow, M. Doring, and A. Wolisz, “Resfi: A secure framework for self organized radio resource management in residential wifi networks,” in *IEEE 17th International Symposium on A World of Wireless (WoWMoM)*, 2016, pp. 1–11.

[33] J.-Q. Li, F. R. Yu, G. Deng, C. Luo, Z. Ming, and Q. Yan, “Industrial internet: A survey on the enabling technologies, applications, and challenges,” *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1504–1526, 2017.

[34] M. Luvisotto, Z. Pang, and D. Dzung, “Ultra high performance wireless control for critical applications: Challenges and directions,” *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1448–1459, 2017.

[35] Y. h. Wei, Q. Leng, S. Han, A. K. Mok, W. Zhang, and M. Tomizuka, “Rt-wifi: Real-time high-speed communication protocol for wireless cyber-physical control applications,” in *IEEE 34th Real-Time Systems Symposium (RTSS)*, 2013, pp. 140–149.

[36] J. Silvestre-Blanes, L. Almeida, R. Marau, and P. Pedreiras, “Online qos management for multimedia real-time transmission in industrial networks,” *IEEE Transactions on Industrial Electronics*, vol. 58, no. 3, pp. 1061–1071, 2011.

[37] “CYW43907: WICED IEEE 802.11 a/b/g/n SoC with an Embedded Applications Processor.” [Online]. Available: http://www.cypress.com/file/298236/download

[38] “BCM4343: Single-Chip 802.11 b/g/n MAC/Baseband/Radio.” [Online]. Available: http://www.cypress.com/file/298081/download

[39] S.-L. Tsao and C.-H. Huang, "A survey of energy efficient mac protocols for ieee 802.11 wlan," *Computer Communications*, vol. 34, no. 1, pp. 54–67, 2011.

[40] V. P. Mhatre, K. Papagiannaki, and F. Baccelli, "Interference mitigation through power control in high density 802.11 wlans," in *INFOCOM*, 2007, pp. 535–543.

[41] A. Kashyap, U. Paul, and S. R. Das, "Deconstructing interference relations in wifi networks," in *SECON*, June 2010, pp. 1–9.

[42] S. Gollakota, F. Adib, D. Katabi, and S. Seshan, "Clearing the RF Smog: Making 802.11 Robust to Cross-Technology Interference," in *ACM SIGCOMM*, 2011, pp. 170–181.

[43] K. Pelechrinis, I. Broustis, S. V. Krishnamurthy, and C. Gkantsidis, "A measurement-driven anti-jamming system for 802.11 networks," *IEEE/ACM Transactions on Networking*, vol. 19, no. 4, pp. 1208–1222, 2011.

[44] A. Zanella, "Best practice in rss measurements and ranging," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2662–2686, 2016.

[45] B. Dezfouli, M. Radi, S. A. Razak, H.-P. Tan, and K. A. Bakar, "Modeling low-power wireless communications," *Journal of Network and Computer Applications*, vol. 51, pp. 102–126, may 2015.

[46] D. Giustiniano, D. Malone, D. J. Leith, and K. Papagiannaki, "Measuring transmission opportunities in 802.11 links," *IEEE/ACM Trans. Netw.*, vol. 18, no. 5, pp. 1516–1529, 2010.

[47] L. Deek, E. Garcia-Villegas, E. Belding, S.-J. Lee, and K. Almeroth, "Joint rate and channel width adaptation for 802.11 mimo wireless networks," in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2013 10th Annual IEEE Communications Society Conference on*. IEEE, 2013, pp. 167–175.

[48] A. Faridi, B. Bellalta, and A. Checco, "Analysis of dynamic channel bonding in dense networks of wlans," *IEEE Transactions on Mobile Computing*, vol. 16, no. 8, pp. 2118–2131, 2017.

[49] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann, and T. Vazao, "Towards programmable enterprise wlans with odin," in *HotSDN*, August 2012, pp. 115–120.

[50] J. Vestin and A. Kassler, "Qos enabled wifi mac layer processing as an example of a nfv service," in *NetSoft*, April 2015, pp. 1–9.

[51] K.-C. Ting, F.-C. Kuo, B.-J. Hwang, H. Wang, and C.-C. Tseng, "A power-saving and robust point coordination function for the transmission of voip over 802.11," in *International Symposium on Parallel and Distributed Processing with Applications (ISPA)*. IEEE, 2010, pp. 283–289.

[52] Z. Zeng, Y. Gao, and P. Kumar, "Sofa: A sleep-optimal fair-attention scheduler for the power-saving mode of wlans," in *ICDCS*. IEEE, 2011, pp. 87–98.

[53] L. Liu, X. Cao, Y. Cheng, and Z. Niu, "Energy-efficient sleep scheduling for delay-constrained applications over wlans," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 5, pp. 2048–2058, 2014.

[54] B. Dezfouli, M. Radi, and O. Chipara, "Mobility-aware real-time scheduling for low-power wireless networks," in *35th Annual IEEE International Conference on Computer Communications (INFOCOM)*, 2016, pp. 1–9.

[55] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "Nox: towards an operating system for networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 3, pp. 105–110, 2008.

[56] "Floodlight." [Online]. Available: http://www.projectfloodlight.org/floodlight

[57] D. Levin, A. Wundsam, B. Heller, N. Handigol, and A. Feldmann, "Logically centralized? State distribution trade-offs in software defined networks," *HotSDN*, pp. 1–6, 2012.

[58] S. H. Yeganeh, A. Tootoonchian, and Y. Ganjali, "On scalability of software-defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 136–141, 2013.

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

[60] S. Waldbusser, M. Rose, J. Case, and K. McCloghrie, "Protocol operations for version 2 of the simple network management protocol (SNMPv2)." [Online]. Available: https://tools.ietf.org/html/rfc1448

[61] A. Wang, M. Iyer, R. Dutta, G. N. Rouskas, and I. Baldine, "Network virtualization: Technologies, perspectives, and frontiers," *Journal of Lightwave Technology*, vol. 31, no. 4, pp. 523–537, 2013.

[62] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "Flowvisor: A network virtualization layer," *OpenFlow Switch Consortium, Tech. Rep*, vol. 1, p. 132, 2009.

[63] W. Zhou, L. Li, M. Luo, and W. Chou, "REST API design patterns for SDN northbound API," in *WAINA*. IEEE, 2014, pp. 358–365.

[64] L. Richardson and S. Ruby, *RESTful web services*. O'Reilly Media, Inc., 2008.

[65] N. Foster, R. Harrison, M. J. Freedman, C. Monsanto, J. Rexford, A. Story, and D. Walker, "Frenetic: A network programming language," in *ACM Sigplan Notices*, vol. 46, no. 9. ACM, 2011, pp. 279–291.

[66] J. Reich, C. Monsanto, N. Foster, J. Rexford, and D. Walker, "Modular sdn programming with pyretic," *Technical Report of USENIX*, 2013.

[67] C. Monsanto, N. Foster, R. Harrison, and D. Walker, "A compiler and run-time system for network programming languages," in *ACM SIGPLAN Notices*, vol. 47, no. 1. ACM, 2012, pp. 217–230.

[68] C. Trois, M. D. Del Fabro, L. C. E. de Bona, and M. Martinello, "A Survey on SDN Programming Languages: Toward a Taxonomy," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2687–2712, 2016.

[69] J. Medved, R. Varga, A. Tkacik, and K. Gray, "Opendaylight: Towards a model-driven sdn controller architecture," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on a*. IEEE, 2014, pp. 1–6.

[70] J. D. Case, M. Fedor, M. L. Schoffstall, and J. Davin, "Simple network management protocol (snmp)," Tech. Rep., 1990.

[71] D. Harrington, R. Presuhn, and B. Wijnen, "RFC 3411: An architecture for describing simple network management protocol (SNMP) management frameworks," 2002. [Online]. Available: https://tools.ietf.org/html/rfc3411

[72] W. Stallings, *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2*. Addison-Wesley Longman Publishing Co., Inc., 1998.

[73] A. Clemm, *Network management fundamentals*. Cisco Press, 2006.

[74] M. K. Kona and C.-Z. Xu, "A framework for network management using mobile agents," in *ipdps*, 2002.

[75] F. A. Silva, L. B. Ruiz, T. R. M. Braga, J. M. S. Nogueira, and A. A. F. Loureiro, "Defining a wireless sensor network management protocol." in *LANOMS*, 2005, pp. 39–50.

[76] R. Enns, M. Bjorklund, and J. Schoenwaelder, "Network configuration protocol (netconf)," 2011. [Online]. Available: https://tools.ietf.org/html/rfc6241.html

[77] D. Stanley, P. Calhoun, and M. Montemurro, "Control and provisioning of wireless access points (CAPWAP) protocol specification." [Online]. Available: https://tools.ietf.org/html/rfc5415

[78] N. Modadugu and E. Rescorla, "Datagram transport layer security (DTLS)." [Online]. Available: https://tools.ietf.org/html/rfc4347

[79] C. Shao, H. Deng, R. Pazhyannur, F. Bari, R. Zhang, and S. Matsushima, "IEEE 802.11 Medium Access Control (MAC) Profile for Control and Provisioning of Wireless Access Points (CAPWAP)," Tech. Rep., 2015.

[80] Broadband Forum, "CPE WAN Management Protocol (CWMP)." [Online]. Available: https://www.broadband-forum.org/technical/download/TR-069.pdf

[81] J. Swetina, G. Lu, P. Jacobs, F. Ennesser, and J. Song, "Toward a standardized common m2m service layer platform: Introduction to onem2m," *IEEE Wireless Communications*, vol. 21, no. 3, pp. 20–26, 2014.

[82] A. Al-Fuqaha, A. Khreishah, M. Guizani, A. Rayes, and M. Mohammadi, "Toward better horizontal integration among iot services," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 72–79, 2015.

[83] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow *et al.*, "ONOS: towards an open, distributed SDN OS," in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 1–6.

[84] "Ryu SDN Framework." [Online]. Available: http://osrg.github.io/ryu

[85] J. Smith, J. Woodhams, and R. Marg, *Controller-Based Wireless LAN Fundamentals: An end-to-end reference guide to design, deploy, manage, and secure 802.11 wireless networks*. Cisco Press, 2011.

[86] R. Suri, N. C. Winget, M. Williams, S. Hares, B. O'Hara, and S. Kelly, "Lightweight Access Point Protocol (LWAPP)." [Online]. Available: https://tools.ietf.org/html/rfc5412

[87] "Radio resource management (RRM) under unified wireless networks." [Online]. Available: http://www.cisco.com/c/en/us/support/docs/wireless-mobility/wireless-lan-wlan/71113-rrm-new.html

[88] P. Zerfos, G. Zhong, J. Cheng, H. Luo, S. Lu, and J. J.-R. Li, "Dirac: A software-based wireless router system," in *MobiCom*, September 2003, pp. 230–244.

[89] R. Murty, J. Padhye, A. Wolman, and M. Welsh, "An architecture for extensible wireless lans," in *ACM Hotnets*, 2008, pp. 79–84.

[90] R. Murty, J. Padhye, A. Wolman, and M. Welsh, "Dyson: An architecture for extensible wireless lans," in *USENIX*, June 2010.

[91] R. Chandra, J. Padhye, A. Wolman, and B. Zill, “A location-based management system for enterprise wireless lans,” in *NSDI*, April 2007.

[92] M. Yan, J. Casey, P. Shome, A. Sprintson, and A. Sutton, “ætherflow: Principled wireless support in sdn,” in *IEEE 23rd International Conference on Network Protocols (ICNP)*, 2015, pp. 432–437.

[93] “CPqD SoftSwitch.” [Online]. Available: https://github.com/CPqD/ofsoftswitch13

[94] “OpenWRT.” [Online]. Available: https://openwrt.org

[95] “Hostapd.” [Online]. Available: https://w1.fi/hostapd/

[96] G. Bhanage, D. Vete, I. Seskar, and D. Raychaudhuri, “Splitap: Leveraging wireless network virtualization for flexible sharing of wlans,” in *GLOBECOM*, December 2010.

[97] A. Zubow, S. Zehl, and A. Wolisz, “BIGAP - Seamless handover in high performance enterprise IEEE 802.11 networks,” in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2016, pp. 445–453.

[98] G. Smith, A. Chaturvedi, A. Mishra, and S. Banerjee, “Wireless virtualization on commodity 802.11 hardware,” in *Proceedings of the second ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*. ACM, 2007, pp. 75–82.

[99] K.-K. Yap, R. Sherwood, M. Kobayashi, T.-Y. Huang, M. Chan, N. Handigol, N. McKeown, and G. Parulkar, “Blueprint for introducing innovation into wireless mobile networks,” in *Second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures (VISA10)*, Septemebr 2010, pp. 25–32.

[100] K.-K. Yap, M. Kobayashi, R. Sherwood, T.-Y. Huang, M. Chan, N. Handigol, and N. McKeown, “Openroads: Empowering research in mobile networks,” *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, pp. 125–126, 2010.

[101] T. Nagai and H. Shigeno, “A framework of ap aggregation using virtualization for high density wlans,” in *INCoS*. IEEE, 2011, pp. 350–355.

[102] J. Schulz-Zander, P. L. Suresh, N. Sarrar, A. Feldmann, T. Hühn, and R. Merz, “Programmatic orchestration of wifi networks.” in *USENIX*, 2014, pp. 347–358.

[103] R. Riggio, C. Sengul, L. Suresh, J. Schulzzander, and A. Feldmann, “Thor: Energy programmable wifi networks,” in *IEEE Conference on Computer Communications Workshops*, April 2013, pp. 21–22.

[104] “Odin.” [Online]. Available: http://sdn.inet.tu-berlin.de

[105] V. Sivaraman, T. Moors, H. H. Gharakheili, D. Ong, J. Matthews, and C. Russell, “Virtualizing the access network via open apis,” in *CoNEXT*, December 2013, pp. 31–42.

[106] “Capsulator.” [Online]. Available: https://github.com/peymank/Capsulator

[107] R. Riggio, T. Rasheed, and F. Granelli, “Empower: A testbed for network function virtualization research and experimentation,” in *IEEE SDN for Future Networks and Services (SDN4FNS)*, November 2013, pp. 1–5.

[108] “Energino project.” [Online]. Available: http://www.energino-project.org

[109] “EmPOWER.” [Online]. Available: http://empower.create-net.org/

[110] D. Zhao, M. Zhu, and M. Xu, “Sdwlan: A flexible architecture of enterprise wlan for client-unaware fast ap handoff,” in *International Conference on Computing Communication and Networking Technologies (ICCCNT)*, July 2014, pp. 1–6.

[111] D. Zhao, M. Zhu, and M. Xu, “Supporting one big ap illusion in enterprise wlan: An sdn-based solution,” in *6th International Conference on Wireless Communications and Signal Processing (WCSP)*, October 2014, pp. 1–6.

[112] Y. Yiakoumis, M. Bansal, A. Covington, J. van Reijendam, S. Katti, and N. McKeown, “Behop: A testbed for dense wifi networks,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 18, no. 3, pp. 71–80, 2015.

[113] K. Nakauchi, Y. Shoji, and N. Nishinaga, “Airtime-based resource control in wireless lans for wireless network virtualization,” in *ICUFN*. IEEE, 2012, pp. 166–169.

[114] K. Nakauchi and Y. Shoji, “Wifi network virtualization to control the connectivity of a target service,” *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 308–319, 2015.

[115] J. Petazzoni, “Build reliable traceable, distributed systems with zeromq (zerorpc).” [Online]. Available: http://pycon-2012-notes.readthedocs.org/en/latest/-dotcloudzerorpc.html

[116] “Zeromq.” [Online]. Available: http://zeromq.org/

[117] J. Schulz-Zander, N. Sarrar, and S. Schmid, “Aeroflux: A near-sighted controller architecture for software-defined wireless networks,” in *Open Networking Summit (ONS)*, 2014.

[118] J. Schulz-Zander, N. Sarrar, and S. Schmid, “Towards a scalable and near-sighted control plane architecture for wifi sdns,” in *third workshop on Hot topics in software defined networking (HotSDN)*, 2014, pp. 217–218.

[119] 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 *MobiCom*, 2009, pp. 297–308.

[120] G. Bianchi and I. Tinnirello, “Remarks on ieee 802.11 dcf performance analysis,” *IEEE communications letters*, vol. 9, no. 8, pp. 765–767, 2005.

[121] H. Moura, G. V. C. Bessa, M. A. M. Vieira, and D. F. Macedo, “Ethanol: Software defined networking for 802.11 wireless networks,” in *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, May 2015, pp. 388–396.

[122] “Pantou.” [Online]. Available: https://www.sdxcentral.com/projects/pantou-openwrt/

[123] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia, “Modeling and performance evaluation of an openflow architecture,” in *Proceedings of the 23rd international teletraffic congress*, 2011, pp. 1–7.

[124] L. Vanbever, J. Reich, T. Benson, N. Foster, and J. Rexford, “Hotswap: Correct and efficient controller upgrades for software-defined networks,” in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013, pp. 133–138.

[125] A. Lara, A. Kolasani, and B. Ramamurthy, “Network innovation using openflow: A survey,” *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 493–512, 2014.

[126] A. Rao, “Towards a software defined network based controller for management of large scale wlan,” Master’s thesis, Indian Institute of Technology Bombay, 2015.

[127] M. Słabicki and K. Grochla, “Performance evaluation of snmp, netconf and cwmp management protocols in wireless network,” in *CECNET*, 2014, p. 377.

[128] T. Nitsche, C. Cordeiro, A. B. Flores, E. W. Knightly, E. Perahia, and J. C. Widmer, “IEEE 802.11ad: directional 60 GHz communication for multi-Gigabit-per-second Wi-Fi,” *IEEE Communications Magazine*, vol. 52, no. 12, pp. 132–141, 2014.

[129] M. Bansal, J. Mehlman, S. Katti, and P. Levis, “OpenRadio: A Programmable Wireless Dataplane,” in *First Workshop on Hot Topics in Software Defined Networks (HotSDN ’12)*, August 2012, pp. 109–114.

[130] M. Bansal, A. Schulman, and S. Katti, “Atomix: A framework for deploying signal processing applications on wireless infrastructure,” in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI ’15)*, May 2015, pp. 173–188.

[131] K. Tan, H. Liu, J. Zhang, Y. Zhang, J. Fang, and G. M. Voelker, “Sora: High-performance software radio using general-purpose multi-core processors,” *Communications of the ACM*, vol. 54, no. 1, pp. 99–107, January 2011.

[132] A. Cidon, K. Nagaraj, S. Katti, and P. Viswanath, “Flashback: Decoupled lightweight wireless control,” in *ACM SIGCOMM’12 conference on Applications, technologies, architectures, and protocols for computer communication*, August 2012, pp. 223–234.

[133] “IEEE Std 802.11,” 2016. [Online]. Available: https://standards.ieee.org/findstds/standard/802.11-2016.html

[134] K. Guo, S. Sanadhya, and T. Woo, “Vifi: Virtualizing wlan using commodity hardware,” in *9th ACM workshop on Mobility in the evolving internet architecture*, September 2014, pp. 25–30.

[135] Alliance, NGMN, “5G white paper,” 2015. [Online]. Available: http://www.ngmn.de/fileadmin/ngmn/content/downloads/Technical/2015/NGMN_5G_White_Paper_V1_0.pdf

[136] J. Vestin and A. Kassler, “QoS Management for WiFi MAC Layer Processing in the Cloud: Demo Description,” in *Proceedings of the 11th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, 2015, pp. 173–174.

[137] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, “A software defined networking architecture for the internet-of-things,” *IEEE/IFIP Network Operations and Management Symposium: Management in a Software Defined World*, pp. 1–9, 2014.

[138] X. Jing, A. P. Patil, A. Liu, and D. N. Nguyen, “Multi-tier wireless home mesh network with a secure network discovery protocol,” 2016, US Patent 9,444,639.

[139] M. Yu, L. Jose, and R. Miao, “Software defined traffic measurement with opensketch,” in *10th USENIX conference on Networked Systems Design and Implementation*, April 2013, pp. 29–42.

[140] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "Spotfi: Decimeter level localization using wifi," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 5, pp. 269–282, 2015.

[141] I. Papanikos and M. Logothetis, "A study on dynamic load balance for ieee 802.11b wireless lan," in *COMCON*, 2001.

[142] "IEEE Std 802.11," 2007. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4248378

[143] K. Sinha, S. C. Ghosh, and B. P. Sinha, *Wireless Networks and Mobile Computing*. CRC Press, 2015.

[144] S. Pack, J. Choi, T. Kwon, and Y. Choi, "Fast-handoff support in ieee 802.11 wireless networks," *IEEE Communications Surveys & Tutorials*, vol. 9, no. 1, pp. 2–12, 2007.

[145] "Cisco compatible extension." [Online]. Available: https://www.cisco.com/web/partners/downloads/765/ccx/Comp_Ext_Cust_Preso.pdf

[146] P. Dely, A. Kassler, L. Chow, N. Bambos, N. Bayer, H. Einsiedler, and C. Peylo, "Best-ap: Non-intrusive estimation of available bandwidth and its application for dynamic access point selection," *Computer Communications*, vol. 39, pp. 78–91, February 2014.

[147] Y. Han and K. Yang, "An adaptive mobility manager for software-defined enterprise wlans," in *Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*, 2016, pp. 888–893.

[148] K. Sui, M. Zhou, D. Liu, M. Ma, D. Pei, Y. Zhao, Z. Li, and T. Moscibroda, "Characterizing and improving wifi latency in large-scale operational networks," in *Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2016, pp. 347–360.

[149] T. K. Ho, "Random decision forests," in *Third International Conference on Document Analysis and Recognition (ICDAR '95)*, 1995, pp. 278–282.

[150] Y. Bejerano, S. j. Han, and L. E. Li, "Fairness and load balancing in wireless lans using association control," in *10th Annual International Conference on Mobile Computing and Networking*, 2004, pp. 315–329.

[151] Y. Bejerano, S. J. Han, and L. Li, "Fairness and load balancing in wireless lans using association control," *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, pp. 560–573, 2007.

[152] W. Li, S. Wang, Y. Cui, X. Cheng, R. Xin, M. A. Al-Rodhaan, and A. Al-Dhelaan, "Ap association for proportional fairness in multirate wlans," *IEEE/ACM Transactions on Networking*, vol. 22, no. 1, pp. 191–202, 2014.

[153] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," in *21th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2003, pp. 836–843.

[154] G. Tan and J. Guttag, "Time-based fairness improves performance in multi-rate wlans," in *Annual Conference on USENIX Annual Technical Conference*, 2004, pp. 23–36.

[155] "OMNeT++." [Online]. Available: http://www.omnetpp.org

[156] L. E. Li, M. Pal, and Y. R. Yang, "Proportional fairness in multi-rate wireless lans," in *The 27th Conference on Computer Communications (INFOCOM '08)*, April 2008, pp. 1004–1012.

[157] D. Gong and Y. Yang, "On-line ap association algorithms for 802.11n wlans with heterogeneous clients," *IEEE Transactions on Computers*, vol. 63, no. 11, pp. 2772–2786, 2014.

[158] P. B. Oni and S. D. Blostein, "Ap association optimization and cca threshold adjustment in dense wlans," in *IEEE Globecom Workshops*, December 2015, pp. 1–6.

[159] T. S. Rappaport, *Wireless communications: principles and practice*. New Jersey: prentice hall PTR, 1996, vol. 2.

[160] G. A. Mills-Tettey, A. Stentz, and M. B. Dias, "The dynamic hungarian algorithm for the assignment problem with changing costs," Robotics Institute, School of Computer Science, Carnegie Mellon University, PA, USA, Tech. Rep. CMU-RI-TR-07-27, 2007.

[161] L. Yang, Y. Cui, H. Tang, and S. Xiao, "Demand-aware load balancing in wireless lans using association control," in *IEEE Global Communications Conference (GLOBECOM '15)*, December 2015, pp. 1–6.

[162] F. Xu, X. Zhu, C. C. Tan, Q. Li, G. Yan, and J. Wu, "Smartassoc: Decentralized access point selection algorithm to improve throughput," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 12, pp. 2482–2491, 2013.

[163] W. Wong, A. Thakur, and S. H. G. Chan, "An approximation algorithm for ap association under user migration cost constraint," in *35th Annual IEEE International Conference on Computer Communications (INFOCOM '16)*, April 2016, pp. 1–9.

[164] "NS-3." [Online]. Available: https://www.nsnam.org/

[165] L.-H. Yen, J.-J. Li, and C.-M. Lin, "Stability and fairness of native ap selection games in ieee 802.11 access networks," in *7th International Conference on Wireless and Optical Communications Networks (WOCN)*, 2010, pp. 1–5.

[166] J. Chen, B. Liu, H. Zhou, Q. Yu, G. Lin, and X. Shen, "Qos-driven efficient client association in high-density software defined wlan," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7372 – 7383, 2017.

[167] G. Calinescu, C. Chekuri, M. Pal, and J. Vondrak, "Maximizing a submodular set function subject to a matroid constraint," in *International Conference on Integer Programming and Combinatorial Optimization*, 2007, pp. 182–196, lecture Notes in Computer Science.

[168] X. Zhe, F. Zhang, W. Wang, and S. He, "A polynomial time algorithm for minimizing a nondecreasing supermodular set function and its performance guarantee," *International Journal of Pure and Applied Mathematics*, vol. 52, no. 5, pp. 637–643, 2009.

[169] T. Bonald, L. Massoulié, A. Proutière, and J. Virtamo, "A queueing analysis of max-min fairness, proportional fairness and balanced fairness," *Queueing Systems*, vol. 53, no. 1, pp. 65–84, 2006.

[170] D. Gong and Y. Yang, "Ap association in 802.11n wlans with heterogeneous clients," in *IEEE INFOCOM '12*, March 2012, pp. 1440–1448.

[171] B. Bellalta, A. Checco, A. Zocca, and J. Barcelo, "On the interactions between multiple overlapping wlans using channel bonding," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 2, pp. 796–812, 2016.

[172] I.-G. Lee and M. Kim, "Interference-aware self-optimizing wi-fi for high efficiency internet of things in dense networks," *Computer Communications*, vol. 89-90, pp. 60–74, 2016.

[173] K. Sood, S. Yu, and Y. Xiang, "Software-defined wireless networking opportunities and challenges for internet-of-things: A review," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 453–463, 2016.

[174] D. Singh, G. Tripathi, and A. J. Jara, "A survey of Internet-of-Things: Future vision, architecture, challenges and services," in *IEEE World Forum on Internet of Things (WF-IoT)*, 2014, pp. 287–292.

[175] G. Gibb, G. Varghese, M. Horowitz, and N. McKeown, "Design principles for packet parsers," in *Proceedings of the ninth ACM/IEEE symposium on Architectures for networking and communications systems*, 2013, pp. 13–24.

[176] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "A roadmap for traffic engineering in sdn-openflow networks," *Computer Networks*, vol. 71, pp. 1–30, 2014.

[177] B. Ng, M. Hayes, and W. K. Seah, "Developing a traffic classification platform for enterprise networks with sdn: Experiences & lessons learned," in *IFIP*. IEEE, 2015, pp. 1–9.

[178] S. Yoon, T. Ha, S. Kim, and H. Lim, "Scalable Traffic Sampling using Centrality Measure on Software-Defined Networks," *IEEE Communications Magazine - special issue on Traffic Measurements for Cyber Security*, no. July, pp. 43–49, 2017.

[179] M. Sinky, A. Dhamodaran, B. Lee, and J. Zhao, "Analysis of H.264 Bitstream Prioritization for Dual TCP/UDP Streaming of HD video over WLANs," in *CCNC*, 2015, pp. 576–581.

[180] T. Hobfeld, R. Schatz, M. Varela, and C. Timmerer, "Challenges of QoE management for cloud applications," *IEEE Communications Magazine*, vol. 50, no. 4, pp. 28–36, 2012.

[181] A. Gudipati, D. Perry, L. E. Li, and S. Katti, "Softran: Software defined radio access network," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013, pp. 25–30.

[182] M. Achanta, "Method and apparatus for least congested channel scan for wireless access points," 2006, US Patent No. 20060072602.

[183] A. Mishra, S. Banerjee, and W. Arbaugh, "Weighted coloring based channel assignment for wlans," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 3, pp. 19–31, 2005.

[184] "NS-2." [Online]. Available: https://www.isi.edu/nsnam/ns/

[185] A. Mishra, V. Brik, S. Banerjee, A. Srinivasan, and W. Arbaugh, "A client-driven approach for channel management in wireless lans," in *25th IEEE International Conference on Computer Communications (INFOCOM)*, April 2006, pp. 1–12.

[186] P. San Segundo, "A new DSATUR-based algorithm for exact vertex coloring," *Computers & Operations Research*, vol. 39, no. 7, pp. 1724–1733, 2012.

[187] M. Seyedebrahimi, F. Bouhafs, A. Raschell, M. Mackay, and Q. Shi, "Sdn-based channel assignment algorithm for interference management in dense wi-fi networks," in *European Conference on Networks and Communications (EuCNC)*, June 2016, pp. 128–132.

[188] "Wi-5." [Online]. Available: http://www.wi5.eu/

[189] X. Ling and K. L. Yeung, "Joint access point placement and channel assignment for 802.11 wireless lans," *IEEE Transactions on Wireless Communications*, vol. 5, no. 10, pp. 2705–2711, 2006.

[190] R. Jain, D. M. Chiu, and W. Hawe, “A quantitative measure of fairness and discrimination for resource allocation in shared computer systems,” DEC-Research Report TR-301, Tech. Rep., 1984.

[191] E. Rozner, Y. Mehta, A. Akella, and L. Qiu, “Traffic-aware channel assignment in enterprise wireless lans,” in *IEEE International Conference on Network Protocols*, October 2007, pp. 133–143.

[192] L. Vandenberghe and S. Boyd, “Semidefinite programming,” *SIAM Review*, vol. 38, no. 1, pp. 49–95, 1996.

[193] H. Zhang, H. Ji, and W. Ge, “Channel assignment with fairness for multi-ap wlan based on distributed coordination function,” in *IEEE Wireless Communications and Networking Conference (WCNC)*, March 2011, pp. 392–397.

[194] K. Duffy, D. Malone, and D. J. Leith, “Modeling the 802.11 distributed coordination function in non-saturated conditions,” *IEEE communications letters*, vol. 9, no. 8, pp. 715–717, 2005.

[195] B. A. H. S. Abeysekera, K. Ishihara, Y. Inoue, and M. Mizoguchi, “Network-controlled channel allocation scheme for ieee 802.11 wireless lans: Experimental and simulation study,” in *IEEE 79th Vehicular Technology Conference (VTC Spring)*, 2014, pp. 1–5.

[196] B. A. H. S. Abeysekera, M. Matsui, Y. Asai, and M. Mizoguchi, “Network controlled frequency channel and bandwidth allocation scheme for ieee 802.11a/n/ac wireless lans: Ratop,” in *IEEE 25th Annual International Symposium on Personal Indoor and Mobile Radio Communication (PIMRC)*, 2014, pp. 1041–1045.

[197] Y. Zhang, C. Jiang, Y. Wang, J. Yuan, and J. Cao, “United channel assignments in residential environments,” in *IEEE Global Communications Conference (GLOBECOM)*, December 2015, pp. 1–6.

[198] M. F. Tuysuz and H. A. Mantar, “Smart channel scanning with minimized communication interruptions over IEEE 802.11 WLANs,” in *IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2013, pp. 2218–2222.

[199] S. Jin, M. Choi, L. Wang, and S. Choi, “Fast scanning schemes for IEEE 802.11 WLANs in virtual AP environments,” *Computer Networks*, vol. 55, no. 10, pp. 2520–2533, 2011.

[200] J. Manweiler, N. Santhapuri, R. R. Choudhury, and S. Nelakuditi, “Predicting length of stay at wifi hotspots,” in *INFOCOM*. IEEE, 2013, pp. 3102–3110.

[201] Q. Dong and W. Dargie, “Evaluation of the reliability of rssi for indoor localization,” in *ICWCUCA*. IEEE, 2012, pp. 1–6.

[202] W. Xu, C. Hua, and A. Huang, “Channel Assignment and User Association Game in Dense 802.11 Wireless Networks,” in *ICC*, 2011, pp. 1–5.

[203] R. Zheng and C. Hua, *Channel Selection and User Association in WiFi Networks*. Springer International Publishing, 2016, pp. 103–115.

[204] W. Zhao, H. Nishiyama, Z. Fadlullah, N. Kato, and K. Hamaguchi, “DAPA: Capacity Optimization in Wireless Networks Through a Combined Design of Density of Access Points and Partially Overlapped Channel Allocation,” *IEEE Transactions on Vehicular Technology*, vol. 65, pp. 3715–3722, May 2016.

[205] S. Zvanovec, P. Pechac, and M. Klepal, “Wireless lan networks design: site survey or propagation modeling?” *Radioengineering*, vol. 12, no. 4, pp. 42–49, 2003.

[206] W. Li, Y. Cui, X. Cheng, M. A. Al-Rodhaan, and A. Al-Dhelaan, “Achieving proportional fairness via ap power control in multi-rate wlans,” *IEEE Transactions on Wireless Communications*, vol. 10, no. 11, pp. 3784–3792, 2011.

[207] Y. Bejerano and S.-J. Han, “Cell breathing techniques for load balancing in wireless lans,” *IEEE Transactions on Mobile Computing*, vol. 8, no. 6, pp. 735–749, 2009.

[208] S. Wang, J. Huang, X. Cheng, and B. Chen, “Coverage adjustment for load balancing with an ap service availability guarantee in wlans,” *Wireless networks*, vol. 20, no. 3, pp. 475–491, 2014.

[209] S. Huang, X. Liu, and Z. Ding, “Distributed power control for cognitive user access based on primary link control feedback,” in *INFOCOM*, 2010, pp. 1–9.

[210] Y. Zeng, P. H. Pathak, and P. Mohapatra, “A first look at 802.11ac in action: Energy efficiency and interference characterization,” *IFIP Networking Conference*, pp. 1–9, 2014.

[211] S. Han, X. Zhang, and K. G. Shin, “Fair and Efficient Coexistence of Heterogeneous Channel Widths in Next-Generation Wireless LANs,” *IEEE Transactions on Mobile Computing*, vol. 15, no. 11, pp. 2749–2761, November 2016.

[212] X. Zhang and K. G. Shin, “Adaptive subcarrier nulling: Enabling partial spectrum sharing in wireless lans,” in *ICNP*, 2011, pp. 311–320.

[213] K. Pelechrinis, M. Iliofotou, and S. V. Krishnamurthy, “Denial of Service Attacks in Wireless Networks: The Case of Jammers,” *IEEE Communications Surveys Tutorials*, vol. 13, no. 2, pp. 245–257, 2011.

[214] E. Bayraktaroglu, C. King, X. Liu, G. Noubir, R. Rajaraman, and B. Thapa, “On the performance of ieee 802.11 under jamming,” in *INFOCOM*, 2008, pp. 1265–1273.

[215] K. Pelechrinis, I. Broustis, S. V. Krishnamurthy, and C. Gkantsidis, “Ares: an anti-jamming reinforcement system for 802.11 networks,” in *CoNEXT*, 2009, pp. 181–192.

[216] K. Pelechrinis, C. Koufogiannakis, and S. V. Krishnamurthy, “On the efficacy of frequency hopping in coping with jamming attacks in 802.11 networks,” *IEEE transactions on wireless communications*, vol. 9, no. 10, pp. 3258–3271, 2010.

[217] I. T. Haque and N. Abu-Ghazaleh, “Wireless software defined networking: A survey and taxonomy,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2713–2737, 2016.

[218] D. F. Macedo, D. Guedes, L. F. Vieira, M. A. Vieira, and M. Nogueira, “Programmable networksfrom software-defined radio to software-defined networking,” *IEEE communications surveys & tutorials*, vol. 17, no. 2, pp. 1102–1125, 2015.

[219] M. Richart, J. Baliosian, J. Serrat, and J.-L. Gorricho, “Resource slicing in virtual wireless networks: A survey,” *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 462–476, 2016.

[220] F. Hu, Q. Hao, and K. Bao, “A survey on software-defined network and openflow: From concept to implementation,” *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2181–2206, 2014.

[221] B. Bellalta, L. Bononi, R. Bruno, and A. Kassler, “Next generation ieee 802.11 wireless local area networks: Current status, future directions and open challenges,” *Computer Communications*, vol. 75, pp. 1–25, 2016.

[222] S. Chieochan, E. Hossain, and J. Diamond, “Channel assignment schemes for infrastructure-based 802.11 wlans: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 12, no. 1, pp. 124–136, 2010.

[223] A. Baid and D. Raychaudhuri, “Understanding channel selection dynamics in dense wi-fi networks,” *IEEE Communications Magazine*, vol. 53, no. 1, pp. 110–117, 2015.

[224] E. Charfi, L. Chaari, and L. Kamoun, “Phy/mac enhancements and qos mechanisms for very high throughput wlans: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1714–1735, 2013.

[225] Q. Ni, L. Romdhani, and T. Turletti, “A survey of qos enhancements for ieee 802.11 wireless lan,” *Wireless Communications and Mobile Computing*, vol. 4, no. 5, pp. 547–566, 2004.

[226] J. Choi and K. G. Shin, “Qos provisioning for large-scale multi-ap wlans,” *Ad Hoc Networks*, vol. 10, no. 2, pp. 174–185, 2012.

[227] A. Ozyagci, K. W. Sung, and J. Zander, “Association and deployment considerations in dense wireless lans,” in *IEEE 79th Vehicular Technology Conference (VTC Spring)*, May 2014, pp. 1–5.

[228] L. H. Yen, T. T. Yeh, and K. H. Chi, “Load balancing in ieee 802.11 networks,” *IEEE Internet Computing*, vol. 13, no. 1, pp. 56–64, 2009.

[229] M. Hoyhtya, A. Mammela, M. Eskola, M. Matinmikko, J. Kalliovaara, J. Ojaniemi, J. Suutala, R. Ekman, R. Bacchus, and D. Roberson, “Spectrum Occupancy Measurements: A Survey and Use of Interference Maps,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2386–2414, 2016.

[230] B. Bellalta, “IEEE 802.11ax: High-efficiency WLANs,” *IEEE Wireless Communications*, vol. 23, no. 1, pp. 38–46, 2016.

[231] C. Liang and F. R. Yu, “Wireless network virtualization: A survey, some research issues and challenges,” *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 358–380, 2015.