Witals: AP-centric Health Diagnosis of WiFi Networks

Mukulika Maity*, Bhaskaran Raman†, Mythili Vutukuru†, Avinash Chaurasia†, Rachit Srivastava‡
mukulika,br,mythili,avinashk@cse.iitb.ac.in, rachit.srivastava@airtightnetworks.com
* Department of CSE, IIIT Delhi, India, † Department of CSE, IIT Bombay, India, ‡ Mojo Networks

***Abstract*—In recent years, WiFi has grown in capacity as well as deployment demand. WiFi system administrators (sysads) want a simple answer to the question "Is my WiFi network healthy?", and a possible follow-up "What is wrong with it?", if it is reported as "unhealthy". But we are far from having such an interface today. It is this gap that this work attempts to fill. We present Witals, a system for WiFi performance diagnosis. We first design a causal diagnosis graph, that extensively identifies the underlying cause(s) of WiFi performance problems. Next, we identify a set of metrics corresponding to nodes in this causal graph. These metrics are measured in real-time by an operational AP, and help in quantifying the effect of each cause. We design a diagnosis algorithm based on the causal graph and the metrics, which ultimately presents a sanitized view of WiFi network health to the sysad. We have implemented a prototype of Witals on an enterprise grade 802.11n AP platform. Using a variety of controlled as well as real-life measurements, we show that our diagnosis framework follows ground truth accurately. Witals has also helped the sysads uncover some unexpected diagnoses.**

Index Terms: Health of wireless networks, AP-centric, Causal graph, Metrics

I. Introduction

In the last decade, the raw capacity of WiFi has increased, from a meager 1-2Mbps with pre-802.11b, to several hundreds of Mbps with MIMO-based 802.11n, and few Gbps with 802.11ac. The reach and demand of WiFi deployments have grown too, and so has the variety of clients. However, the system administration related pain-points of WiFi installations have not decreased. Empirical evidence suggests that wireless is considered problematic, even when the problem may lie elsewhere.

As the size and geographic reach of cloud-managed WiFi deployments increase, it is critical for system administrators (sysads) to have a simple interface which tells the health status of a WiFi network: Is the WiFi network healthy? If not, what ails it? While this is an important question, such performance diagnosis is technically challenging due to the multitude of inter-dependent and temporally varying factors. We address this challenge with the design, implementation, and evaluation of Witals, a system for health diagnosis of WiFi installations.

Prior approaches toward such a WiFi performance diagnosis system have been incomplete (details in Sec. II), given the complexity of such diagnosis. Most prior research work either use heavy weight sensor infrastructure ([11], [15]) or extra radio at the AP ([19], [22]) or need client side modifications ([7], [9]). Most of them also use a central analysis engine. But it is hard to justify such additional trace-collection infrastructure for a large fraction of WiFi installations: small and medium businesses, home-office, guest WiFi portals at cafeterias, restaurants, shops, WiFi at point-of-sales and so on. There are some AP-centric diagnosis frameworks ([19], [20], [23]) as well, but these too stop short of diagnosing the root cause(s) of WiFi performance problems. They do not have a framework where multiple simultaneous causes can be quantified or compared. On the other hand, the state-of-the art commercial tools ([3], [5]) provide plots of various metrics (e.g., error rate, bytes downloaded, number of radios in the vicinity, throughput etc.) and use many thresholds. These metrics are incomparable to one another, thus making it further difficult to diagnose root cause(s) of performance problems.

The design of Witals explicitly avoids the requirement of any distributed trace collection, or centralized merging and analysis, instead uses an AP-centric architecture: a live AP is constantly also diagnosing any performance problems present. Witals focuses on WiFi performance problems. Further, we are focused on diagnosis of WiFi performance in the downlink direction, which is the common case.

For such performance diagnosis, we start with a causal diagnosis graph which relates various causes to different kinds of performance problems. The nodes in the graph are possible WiFi performance problems. The edges in the graph are causal relationships between the nodes. The causal diagnosis graph is non-trivial, given the complex interplay of various phenomena at different layers of the network stack, in a WiFi network. Our causal diagnosis graph identifies an extensive set of performance problems (Sec. III). This distilled yet extensive picture of WiFi performance causes is our first contribution, vis-à-vis prior work.

Corresponding to different nodes of the causal diagnosis graph, we identify metrics (witals: WiFi vital statistics) which can be measured real-time in an operational AP (Sec. IV). The set of metrics considered in prior work is vast; e.g., MAC layer throughput, frame loss rate, bit-rate. Some of these metrics affect many nodes in the causal graph, hence, it is not possible to diagnose the root cause(s) from these metrics. On the other hand, some metrics do not affect any node. Further, some of the metrics are not in the same unit as other metrics, for example, average bit-rate with airtime utilization and so on. In contrast, our metrics (a) are directly related to specific nodes of the diagnosis graph, thus helping in diagnosis, (b) *quantify* the extent to which a cause is affecting the overall performance, and importantly (c) they

First author was at IIT Bombay during this work are *comparable* with one another. The metrics being comparable allows the identification and quantification of multiple simultaneous causes of WiFi performance problems, which is a common occurrence. The careful identification of the comparable metrics, each measuring the effect of a cause on throughput loss, is another aspect separating Witals from prior work and current commercial systems; this is our second contribution.

Next, using the witals metrics and the causal diagnosis graph, we present a diagnosis algorithm (Sec. V). The purpose of this algorithm is to present a refined view to the sysad, by representing the health of the WiFi networks over time. Fig. Fig. 1(a) shows the values of the witals metrics for one of our real-life measurements, where 80 clients were associated with an AP. The network was being used in a classroom for an online quiz activity, and the user experience was poor. Note that, state-of-the-art in current commercial systems stops with such a view of a multitude of graphs. Although the metrics are comparable, Fig. 1(a) is not easy to interpret visually. Fig. 1(b) shows output of our Witals algorithm. A bubble in a column indicates that the corresponding column has been diagnosed as a performance problem. Also, the width is proportional to the throughput reduction it causes. As can be seen from Fig. 1(b), Witals, capable of diagnosing multiple simultaneous causes, diagnoses over-crowding, suboptimal rate adaptation algorithm, data and control, management frames of other neighboring BSS and interference from other neighboring BSS as the major reasons for poor performance of this network. The sysad can drill-down this final simplified view for further details, such as to learn the throughput impact of each cause, if and when necessary. The diagnosis algorithm's ability to present a simplified picture to the sysad, even in presence of complex interactions between multiple simultaneous causes, is also a novel aspect of Witals, and forms our third contribution.

We have implemented the Witals framework on an enterprise grade 802.11n AP platform, and used this prototype to evaluate Witals extensively (Sec. VI). Using various controlled experiments as well as measurements in different uncontrolled real-life settings, we show that WiFi performance diagnosis using Witals matches ground-truth effectively. Witals helped the sysads unearth some unexpected root-causes behind performance degradation in certain deployments. We also compare Witals diagnosis with both research and commercial tools quantitatively. The extensive implementation based evaluation is the fourth and final contribution of this work. While Witals performs root-cause diagnosis from a single-AP perspective, most of our evaluation have been in campus/enterprise settings with multiple surrounding APs. We show that this single-AP perspective itself is of significant value, and presents a substantial challenge. In Sec. VII, we discuss how we can extend this framework from multiple AP perspective. In future, we plan to extend Witals framework for such settings.

II. Related Work

Diagnosing the health of wireless networks has been very interesting to the research community and also to the commercial AP vendors. Researchers have proposed different tools to diagnose the potential problems in the wireless networks using various architectures. We categorize them into three main classes: (1) separate infrastructure of monitors/sniffers, (2) client based, and (3) AP-centric architecture. Below we discuss few examples in each category.

1) ***Separate infrastructure of monitors/sniffers****:* Jigsaw [11] and Wit [15] deploy monitors (or WiFi sensors) throughout the network, collect comprehensive link layer traces at each sensor, transfer them centrally, and merge the traces to provide a unified view of the network. The work in [10] builds on Jigsaw and classifies TCP throughput issues, assuming TCP header availability. PIE [22] builds on Jigsaw and Wit, and uses traces from multiple APs and analyses at a central controller. It detects which two specific nodes are interfering in an enterprise network. The implementation uses an additional radio per AP, which is akin to a distributed sensor infrastructure. MOJO [21] also uses a separate infrastructure of sniffers to collect data, and an inference engine to diagnose the problems. This paper focuses on physical layer anomaly detection, specifically the detection of hidden terminal vs capture effect and detection of noise vs signal strength variation. Other performance problems are not considered here.

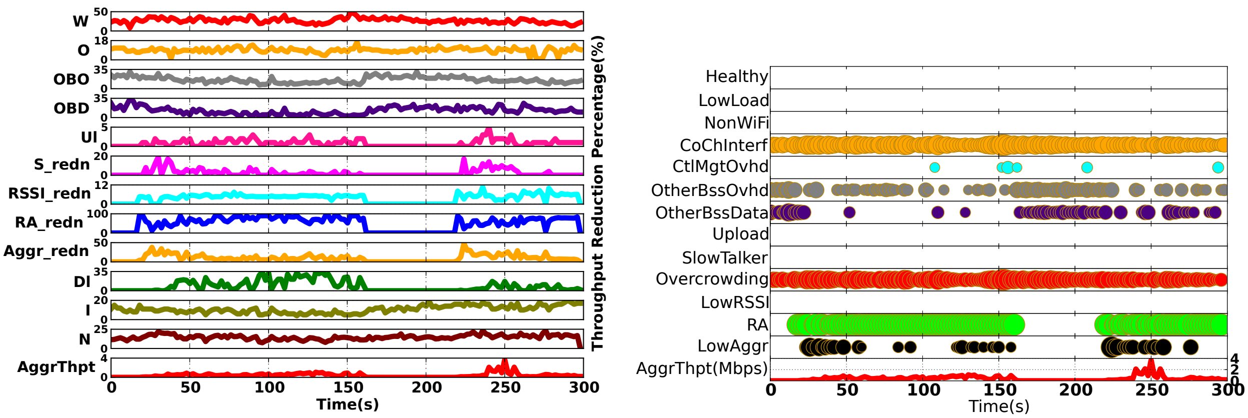
While these systems use separate infrastructure of monitors/sniffers, it might be an overkill for a large fraction of WiFi installations: small and medium businesses, home-office, guest WiFi portals at cafeterias, restaurants, shops, WiFi at point-of-sales and so on. The avoidance of heavy-weight link layer trace collection and merging is of special significance in today's high data rate 802.11n and 802.11ac. Hence, we design Witals to diagnose the health of wireless networks without using any extra infrastructure.

2) ***Client based architecture****:* WiFi-Profiler [9] and Client-Conduit [7] represent early work on helping WiFi clients diagnose problems at the client side. These architectures use help from other nearby clients. In WiMed [13], the diagnosis is done at the client using a second sensing radio. It classifies non-WiFi interference types to: bluetooth, microwave etc..

These architectures, unlike our work Witals, require client side modifications. In a way, these are complementary to Witals, which performs diagnosis at an AP.

3) ***AP-centric architecture****:* These research tools use inline measurement architecture i.e., AP as the vantage point. They collect measurements at the operating AP itself and avoid use of any other infrastructure for trace collection. The work in WTF [23] is interested in distinguishing between wireless faults versus other ISP faults in home networks. But it stops short of diagnosing the root cause of any wireless problem present. Architecturally, it is similar to Witals in the use of an AP-centric measurement framework, but unlike Witals, the fault diagnosis algorithm runs at a central server. WTF also requires TCP header inspection, unlike Witals.

Antler [20] is another system for AP-centric wireless performance diagnosis. It uses a hierarchical approach for collecting metrics. As a baseline operation, Antler collects a minimal set of metrics to represent the health of a wireless network. Once these metrics cross the threshold, the second tier of metrics are collected to have a more detailed view of the network. If second tier of metrics also cross the threshold, then collection of third tier of metrics starts and so on. Like WTF, Antler too requires data to be sent to a central server for



(a) Witals metrics vs time

(b) Witals diagnosis

Fig. 1: Witals metrics and Witals diagnosis for classroom-80, a real-life scenario

diagnosis. As we shall show in Sec. VI, the causal diagnosis in Witals is extensive. On the other hand, Antler cannot diagnose several wireless conditions such as slow talker, sub-optimal rate-adaptation, 802.11n aggregation-related throughput loss and so on. It also cannot distinguish between different causes of interference, which Witals can. Further, it does not provide a framework where the effect of multiple simultaneous causes can be quantified and compared. We have implemented Antler and compared it with Witals in Sec. VI.

WISE [19] is an AP-centric measurement framework for observing home wireless experience. The framework uses primary radio of the AP to collect information about its clients, a second radio for collecting additional measurements and a remote measurement controller for control and configuration of the APs. WISE relates various causes of poor wireless performance and comes up with a new metric Witt i.e., WiFi-based TCP throughput. It predicts the TCP throughput of a new flow in the current wireless condition. WISE computes Witt metric using $link_{exp}$ model, it is $link_{exp} = (1-a)*(1-c)*r, 0 < a < 1, o < c < 1$. Where a is airtime utilization from external sources, c is local contention and r is effective rate. WISE uses Witt metric to classify WiFi performance into 5 categories, ranging from very poor to very good.

WISE mainly focuses on offline analysis. On the contrary, Witals is made to run in real time. Like PIE, WISE too uses an additional radio per AP. Further, WISE needs higher layer packet header inspection, unlike Witals. Also, WISE can not diagnose several wireless conditions for e.g., sub-optimal rate adaptation algorithm, poor efficiency due to low level of frame aggregation etc.. WISE computes local contention from the relative amount of other clients' traffic to the total traffic at the AP. This provides a wrong diagnosis when this ratio is high for other reasons (e.g., slow talker) apart from contention. We have implemented WISE and compared it with Witals as part of evaluation in Sec. VI.

Table I compares prior work with Witals. In summary, none of the prior work addresses WiFi performance diagnosis in an extensive yet simplified manner for a sysad. Nor can they quantify the effect of a cause or diagnose multiple simultaneous causes. In addition, much of the prior work has used 802.11a/b/g for prototyping and evaluation. The Witals prototype uses 802.11n, which is a significant test of its low overhead in real-time data collection and diagnosis.

In terms of commercial systems (e.g., [1], [2], [3], [5]), while AP vendors are very interested in WiFi performance diagnosis, the state of the art is to present a variety of graphs of different metrics (e.g., error rate, bytes downloaded, number of radios in the vicinity, throughput etc.) incomparable to one another. Also, any problem reporting is based on arbitrarily-set thresholds for these different metrics. The incomparable metrics make a sysad's job difficult to diagnose the root cause of WiFi performance problem. We quantitatively compare Witals with popular commercial tools in Sec. VI.

III. Causal Diagnosis Graph

The performance of a WiFi network might be poor for a variety of reasons. We design Witals to examine the health of the wireless networks. Additionally, it can be integrated with other network management solutions which diagnose problems outside a WiFi network. Witals diagnoses the WiFi network's health from the perspective of the downlink performance of clients of the AP on which Witals runs. While uplink performance issues are important too, the focus of this work is diagnosing downlink performance issues. In Sec. VII, we discuss how we can extend this framework to diagnose the uplink performance issues. Henceforth in the paper, we use the term AP_W to refer to the access point on which an instance of Witals is running, while talking about performance diagnosis by that AP.

We first describe the causality between possible performance problems and their effect on the WiFi network. Fig 2 shows the Witals *causal* graph. The nodes are possible WiFi performance problems. The edges are causal relationships between the nodes: $a \rightarrow b$, means a causes b.

Note that none of the causal relationships are novel or unknown; our contribution lies in careful identification and organization of extensive causes of performance problems.

At a high level, if downlink throughput of AP_W is low, there are only three possible reasons A. Low offered load: AP_W does not have enough data to transmit, B. Not enough airtime: AP_W has data to transmit but does not get enough opportunities to transmit, C. Inefficient utilization of provided airtime: AP_W has data to transmit and it gets enough opportunities to transmit as well but it is not able to utilize airtime efficiently. Note that by design, one or more of the above *must* happen, for the observed downlink throughput to be poor. That is, if there is enough offered load, sufficient airtime for the AP,

<table>
<tr><th>Feature</th><th>Jigsaw / Wit</th><th>PIE</th><th>MOJO</th><th>WiFi profiler, Client-Conduit</th><th>WiMed</th><th>WTF</th><th>Antler</th><th>WISE</th><th>Witals</th></tr>
<tr><td>Additional resource for trace collection?</td><td>Yes</td><td>Yes</td><td>Yes</td><td>No</td><td>Yes</td><td>No</td><td>No</td><td>Yes</td><td rowspan="3">No</td></tr>
<tr><td>Additional resource for analysis?</td><td>Yes</td><td>Yes</td><td>Yes</td><td>No</td><td>No</td><td>Yes</td><td>Yes</td><td>Yes</td></tr>
<tr><td>Client side modification?</td><td>No</td><td>No</td><td>No</td><td>Yes</td><td>Yes</td><td>No</td><td>No</td><td>No</td></tr>
<tr><td>Performance issues covered extensively?</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td><td rowspan="3">Yes</td></tr>
<tr><td>Diagnosis & quantification of multiple simultaneous causes?</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td></tr>
<tr><td>802.11n/ac support?</td><td>No</td><td>No</td><td>No</td><td>No</td><td>No</td><td>Yes</td><td>No</td><td>Yes</td></tr>
</table>

TABLE I: Summary of comparison of Witals with prior work

and efficient utilization of the same, downlink throughput is necessarily good.

A. Low offered load

The first possible reason for low throughput, marked as "A" in Fig 2, is low offered load to the wireless network. This could in turn be caused due to a variety of non-wireless related reasons such as wired network bottleneck, overloaded web server, or simply when the wireless clients are not generating enough traffic. Witals does not distinguish between the different reasons causing low offered load.

The next two subsections discuss the remaining two possible reasons for low throughput B. & C..

B. Not enough airtime for AP_W

We now discuss all the possibilities where AP_W has data but does not get enough airtime to transmit: these are shown under the node marked "B" in Fig 2. When AP_W does not get sufficient airtime, this must necessarily be due to one of the following possibilities.

B.1 Wasted non decodable energy: The channel is occupied with energy which is not decodable. Such energy could either be *non-WiFi energy* or *WiFi non-frame energy*. The former could be due to any of a number of non-WiFi devices operating in the same frequency range, e.g., microwave oven, baby monitor, etc. Or it could also be due to WiFi radios far away from AP_W. The latter is co-channel interference: energy which can be detected as WiFi transmission but cannot be decoded into an intelligible frame (e.g., sync loss after PLCP detection). We have observed that with the high data rates of 802.11n, such instances are quite common. When a neighbouring AP transmits at a high data rate (say 300 Mbps) to its nearby client, it is likely that AP_W detects it as WiFi, but cannot decode the bits of the frame: the high data rates of 802.11n are decodable only at short distances (10-30 feet).

B.2 Other BSS: Some of the energy which is decodable, could belong to other BSS in the vicinity, sharing AP_W's channel. Now, this might be caused by data frames of other BSS or control and management frames of other BSS.

B.3 Control, management overheads: Within AP_W's BSS, airtime would be taken up by control and management overheads in WiFi (e.g., beacons, probe request/response, etc.). We also include here, frames received with FCS (Frame Check Sequence) error.

B.4 Upload: Within AP_W's BSS, a portion of the airtime would be used for uplink transmissions, which effectively causes downlink airtime reduction.

B.5 MAC unfairness: Now, even when the above factors are not affecting AP_W's airtime, the distributed operation of the 802.11 DCF MAC can result in short term or even long term unfairness, which can even be severe. Such unfairness can be in terms of (i) transmission opportunities (txop) or (ii) airtime. (i) AP_W may not get sufficient txop when it is the middle node in a classical hidden terminal scenario [14]: AP_W loses the DCF contention with high probability to one of the other two nodes. (ii) Even with fair txop, due to different bit-rates of the transmitters, there can be unfairness in airtime. When a radio operates at a low data rate (say 6Mbps) and occupies most of the time in the channel, then it slows down faster radios (say 144Mbps) significantly. This is the classical slow talker problem.

B.6 Overcrowding: Finally, even when airtime is fair, there may simply be too many radios (N) in the channel. AP_W may get $\frac{1}{N}$ airtime, which would result in less airtime if N is large.

C. Inefficient utilization of provided airtime

In this subsection we discuss all the possibilities when AP_W has data to transmit and gets enough airtime but is not able to utilize airtime efficiently. These are shown as the causes of the node marked "C" in Fig 2.

C.1 Low bit-rate: A straightforward reason for poor utilization of airtime is that AP_W's transmissions are at a low physical layer bit-rate. AP_W may be using a low bit-rate if the client has a low signal strength (RSSI) and thus cannot work at a high bit-rate. Or even if RSSI is good, AP_W's rate adaptation algorithm could be picking a sub-optimal bit-rate. A classical case of this is when the rate adaptation algorithm confuses non-channel losses (e.g., collisions) for channel losses due to low SNR.

C.2 High loss rate: Even when AP_W is operating at an optimal bit-rate, airtime may not be effectively used if there are too many link layer losses leading to retransmissions. There are two possible reasons for high loss rate. First, *DCF collisions:* 802.11 is a contention based access mechanism. When radios pick the same random number for backoff before transmission, collisions happen, the probability of which increases with the number of radios. Second, *any other client side issues:* it might be possible that when AP_W transmits, it may be hidden from some source of WiFi or non-WiFi energy which interferes at the client. Another client-side reason for high loss rate we have observed is that several clients are unable to handle high levels of 802.11n aggregation, losing frames between the wireless hardware and the driver; i.e., the client has a CPU/system bottleneck.

C.3 Low aggregation: In case of 802.11n, even when AP_W is operating at an optimal bit-rate and the loss rate is also low, airtime utilization can be inefficient if level of frame aggregation is low. Prior work [18] has reported this and we have observed the same in our experiments too.

D. Cross-links in the causal graph

So far we have described the causal elements in the causal graph (Fig. 2) from the perspective of insufficient airtime and

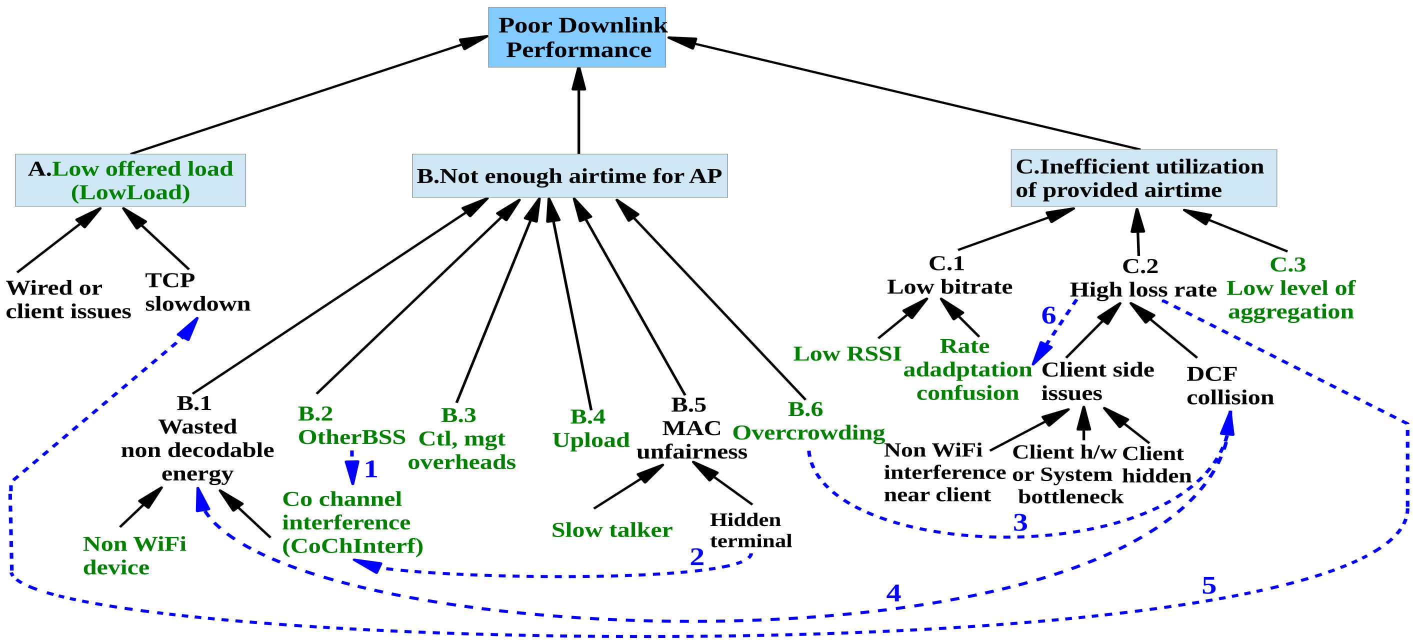


Fig. 2: Witals Diagnosis: Causal Graph (underlined green nodes are diagnoses of Witals)

inefficient usage of provided airtime. But the various nodes in the causal graph (causes) we have introduced have other effects too. We now describe these as cross-links (dashed lines, marked 1-6) in the causal graph.

When other APs (BSS) operate on same or nearby channel of AP_W, airtime reduction for AP_W is one possible effect. If these APs are too far from AP_W or use high data rates, these transmissions will not be decodable to AP_W. This is co-channel interference, and hence shows up as an increase in wasted non-decodable energy. Fig. 2 shows this as **cross-link 1**.

When AP_W is a middle node in a hidden terminal scenario, this results in collisions being seen at AP_W. This too shows up as wasted non-decodable energy. This cause-effect relation is shown as **cross-link 2** in Fig 2.

When there are a large number of transmitters (N) in the channel, not only is the airtime share ($\frac{1}{N}$) of AP_W low, but this also has another important effect. Overcrowding also causes a large number of DCF collisions due to high chance of two nodes picking the same random number for backoff. This is shown as **cross-link 3** in Fig. 2. Further, when two frames collide, the energy becomes non decodable and thus wasted. This is shown as **cross-link 4** in Fig 2.

When there is high loss rate on the channel either due to overcrowding (high collisions and so high frame losses) or client side issues, this interacts poorly with TCP. High loss rate could cause variable RTT for TCP, DUP ACKs, and spurious TCP timeouts (also reported by prior work [17]). Thus TCP slows down in response to these, causing low offered load to WiFi. This interaction is shown by **cross-link 5** in Fig 2.

A more immediate effect of high losses is that the rate adaptation algorithm, unable to distinguish non channel losses from channel losses, reduces the bit-rate. This is shown in Fig 2 as **cross-link 6**.

E. Causal diagnosis, limitations, actions

The causal graph as described above is the basis for the design of the diagnosis algorithm (Sec. V). The underlined green nodes in Fig. 2 are the various conditions Witals can diagnose. By design, the leaf nodes (i.e., those without any incoming edge) of the graph represent the diagnoses of Witals. However, there are two exceptions: (1) some leaf nodes are not associated with any diagnosis, and (2) some non-leaf nodes represent diagnoses. We describe them in more detail below.

(1) As mentioned earlier, a limitation of Witals is that, due to its AP-centric architecture, we cannot diagnose specific client-side issues and wired side issues. Hence, we can not diagnose the leaf nodes associated with client side issues and wired side issues. As discussed before, TCP slowdown is caused by high frame losses (cross-link 5), that in turn is caused by overcrowding. Also, high frame losses cause rate adaptation confusion (cross-link 6). Therefore, in occasion of TCP slow down, overcrowding and RA based problems show up.

Aside from these, a specific wireless related leaf node we do not diagnose is that of hidden terminal. The reason for this is as follows. We attempted to create the MAC unfairness starvation scenario as in [14], experimentally, with AP_W in between two other APs. However, we observed that starvation rarely happens; instead the hidden terminal problem shows up primarily as increase in wasted non-decodable energy, rather than as MAC unfairness. That is, the cross-link is experimentally observed to be stronger than the primary causal effect expected. Hence this hidden terminal scenario manifests itself as co-channel interference, which is what Witals diagnoses. Another possible hidden terminal scenario is between two clients of AP_W, with AP_W as the middle node. Since Witals does not have client-side view, we cannot diagnose this situation. But we expect this to be rare, since the two clients have to be at opposite ends of AP_W's coverage, for this situation to arise.

(2) There are three non leaf nodes that represent diagnoses are: (a) co-channel interference, (b) rate adaptation confusion, and (c) low offered load. (a) Co-channel interference causes increase in wasted non-decodable energy. But co-channel

interference itself is caused by other BSS in the vicinity. Due to this dependency, it becomes a non leaf node, but this is a diagnosis as well. (b) High number of frame losses causes confusion to the rate adaptation algorithm, which lowers the bit-rate in response. Due to this dependency, suboptimal rate adaptation algorithm becomes a non leaf node, but this also is a diagnosis. (c) As discussed before, we can not diagnose wired or client side issues, and TCP slowdown shows up as overcrowding and RA. Therefore, we diagnose the non-leaf node low offered load to indicate idle channel condition.

IV. Witals for Diagnosis

What metrics can we possibly use for diagnosing problems? The spectrum of performance metrics considered in prior works is vast. For example, Antler [20] considers the following metrics: minimum throughput, overhead, airtime utilization of the AP and the clients, signal strength of the clients, data rate, packet loss rate, retransmission percentage. WISE [19] considers the following metrics for computation of $Witt$: airtime utilization, local contention, effective rate. We categorize the set of metrics considered in prior work into following categories. (1) Metrics related to TCP, (2) metrics related to MAC throughput, (3) metrics related to number of WiFi entities, (4) metrics related to number of bytes, (4) metrics related to airtime utilization, (6) metrics related to signal strength (7) metrics related to bit-rate, (8) metrics related to loss rate, (9) metrics related to overhead, (10) metrics related to contention, and (11) metrics related to non-WiFi.

Out of all these metrics, we choose a specific subset. In choosing the metrics (called witals, or WiFi vital statistics), we have the following guidelines. (1) We should be able to diagnose *multiple simultaneous causes* of WiFi performance problems, as this is a common scenario. (2) It should be easy to measure and maintain in real-time in a live AP, even when operating at the high data rates of 802.11n or 802.11ac.

Common unit of measurement: Given the above guidelines, we choose a minimal set of metrics. Importantly, we represent all of our metrics in the same comparable unit of measurement. We choose *percentage reduction in downlink throughput* due to a cause, as this common comparable unit. This is an important choice, as this allows us to compare multiple possible simultaneous causes of performance problems, and flag the major ones.

Time window of measurement: We measure metrics in windows of time, of duration denoted as T_W. The choice of T_W has to be such that it is large enough to tide over the fine grained frame-level behaviour of 802.11 DCF. It also has to be large enough to amortize the fixed overheads of implementing the diagnosis algorithm. T_W also has to be small enough to detect a problem before it "disappears" (e.g., a value of a minute or two is likely to be too high). In practice, we have found that T_W of a few seconds works well.

In Table II, we show the set of metrics considered in prior work. We also indicate whether we choose them in our framework or not, the justification for the same, and the corresponding witals metrics (if any). Some of the metrics such as TCP RTT, MAC throughput considered in prior work affect many nodes in our diagnosis graph, hence, it is not possible to diagnose the root cause(s) from these metrics. Therefore, we do not choose such metrics. On the contrary, some metrics such as bytes downloaded, number of associations are not related to any node, we do not choose them as well. Some other metrics such as signal strength of the AP and the clients, average bit-rate and so on, are not comparable with other metrics such as airtime utilization. Hence, we modify these metrics in an appropriate way to make them comparable with others.

As described earlier, the underlined green nodes in the causal graph of Fig. 2 are various diagnoses of Witals. We come up with a set of metrics, that helps us diagnose the green nodes of the causal graph. The metrics fall into two categories: (a) airtime reduction metrics, (b) metrics representing inefficient utilization of airtime. These mirror the two major sets of causes marked "B" and "C" in the causal graph. And all metrics are represented in terms of percentage of downlink throughput reduction they cause. The close connection between the metrics below and the nodes in the causal graph is captured in Table III. It lists the various possible diagnoses of Witals, each diagnosis corresponds to a node of the causal graph of Fig. 2. An important point is that comparable witals metrics and causal relationships in the causal graph allow the detection of multiple simultaneous problems, which is a common occurrence, especially due to the complex inter-dependences as shown in the causal graph. Table III also shows the possible follow-up actions for each diagnosis. We must note here that the follow-up actions given here are merely suggestive and not exhaustive. While some follow-up actions are short term in nature (e.g., switch AP channel), some are longer term (e.g., re-plan WiFi installation).

A. Airtime reduction metrics

We use the following metrics to measure downlink throughput reduction due to airtime reduction for AP_W.

Wasted non-decodable energy (W): This metric measures the percentage of airtime lost (over the measurement window T_W) due to non-decodable energy. This metric is a direct measure of the throughput reduction attributable to this cause.

To measure W, we have used vendor support in the AP driver. The API provides means to measure the amount of airtime for which there was any energy detected on air. It also provides means to get per-frame information (length, data-rate etc.) for all frames on air, using which we can get the time duration of decodable energy. Subtracting the latter from the former, we get W. We note that the above API is not unique to the vendor's driver, similar ones are available in popular open-source drivers such as ath9k as well as in broadcom chipset based driver.

CtlMgtOverhead (O), OtherBSSOvhd (OBO), OtherBSSData (OBD), Upload (Ul): Once we have per frame (length, data-rate) information, the airtime occupied by control and management overheads (O) of own BSS, control and management overheads (OBO) of other BSS, data frames of other BSS (OBD), upload (Ul) are all straightforward to compute. When represented as a percentage of measurement window T_W, like with W, these form direct measures of throughput reduction due to each of these causes.

Metric	Chosen	Reason	witals
Metrics related to TCP: TCP RTT [23], TCP throughput [2], [1]	No	Many nodes are affected by these metrics, not possible to diagnose root cause(s) from these	-
Metrics related to MAC throughput: average, aggregate, minimum throughput [2], [6]	No	Many nodes are affected by these metrics, not possible to diagnose root cause(s) from these	-
Metrics related to number of WiFi entities: number of associations [5]	No	Not related to any node	-
Metrics related to number of bytes: downloaded/uploaded/total bytes [13]	No	Not related to any node	-
Metrics related to airtime utilization [20], [19]			
Channel utilization	Yes	Direct mapping, used to diagnose low offered load	I
Airtime utilization of the AP	No	Our focus is to maximize this metric	Dl
Airtime utilization of the clients	Yes	Direct mapping, used to diagnose upload	Ul
Metrics related to signal strength: signal strength of the clients [20]	Yes (modified)	The raw signal strength values are not comparable with others, we modify it in an appropriate way to make it comparable	$RSSI_{redn}$
Metrics related to bit-rate: average bit-rate [20], effective rate [19]	Yes (modified)	The bit-rate values are not comparable with others, we modify it in an appropriate way to make it comparable	RA_{redn}
Metrics related to loss rate: frame loss rate [20], frame retx % [1], error rate [19]	No	We use these to determine rate adaptation based problems	-
Metrics related to overheads: control, management overhead [20]	Yes	Direct mapping, used to diagnose control, management overheads	O
Metrics related to contention: local contention [19], percentage of data frames collided [21]	Yes (modified)	These metrics are not comparable with others, we modify them in an appropriate way to make it comparable	W, N
Metrics related to non-WiFi energy: non-WiFi energy [19], RF-interference [1]	Yes	Direct mapping, used to diagnose non-WiFi devices	W

TABLE II: The set of metrics considered in prior work and their applicability in Witals framework

WiFi Diagnosis	Metric	Possible remedial actions
Healthy	-	Good news for WiFi sysad; source of any problem is elsewhere
LowLoad	I	Implies healthy; source of any problem is non-wireless
NonWiFi	W	Short to medium term: look for obvious sources of nonWiFi interference (microwave etc); long term: consider 5Ghz, fewer interfering devices
CoChInterf	W, OBO & OBD	Short to medium term: better channel planning; long term: consider 5Ghz, more channels
CtlMgtOvhd	O	Better configuration of access point
OtherBSSOvhd	OBO	Short to medium term: better configuration of other BSS; long term: consider 5Ghz, more channels
OtherBSSData	OBD	Short to medium term: better channel planning; long term: consider 5Ghz, more channels
Upload	Ul	Consider imposing a limit on upload
SlowTalker	S_{redn}	Own client slow $\Rightarrow$ discourage slow clients at critical times (medium term); implement airtime fairness (long term). Slow talker out of sysad's control $\Rightarrow$ consider switching channel/band of operation
Overcrowding	W, N	If own clients are too many, consider imposing a limit, especially if its a guest WiFi; timed access for guests; if too many BYOD clients stricter BYOD policy; consider deploying more APs.
LowRSSI	$RSSI_{redn}$	Suggest client to move to another location, Re-plan WiFi installation, set user expectation (good coverage guaranteed only at certain locations)
RA	RA_{redn}, N	Use collision aware rate adaptation (e.g CARA [12], YARAa [8]). Investigate client side issues.
LowAggr	$Aggr_{redn}$	Suggest better client hardware or better aggr. algo at AP

TABLE III: Diagnosis Table

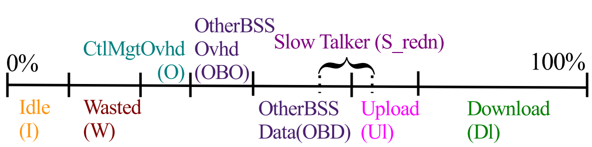


Fig. 3: Break up of airtime

Fig. 3 shows the split of airtime among the above metrics. We also note that we can compute the percentage of idle time I, subtracting out the non-decodable and decodable portions of airtime from the total of 100%.

Slow talker reduction (S_{redn}): A metric whose design is non-trivial is the slow-talker reduction (S_{redn}) metric. Prior work has considered metrics such as ratio of the fast radio to the slower one [22]. Instead, in accordance with our guiding principle that we wish all metrics to be *comparable* to one another, we wish to design S_{redn} to reflect throughput reduction due to the slow talker problem.

Thus, to compute S_{redn}, we ask the question "how much has AP_W been slowed down due to slower radios ?". Or more precisely, "how much more airtime (throughput) could AP_W have achieved had all radios been at least as fast as AP_W ?". Posed in this fashion, the computation of S_{redn} is a sequence of straightforward algebra. We first calculate AP_W's average bit-rate during a T_W, as a ratio of the number of bytes B_{Total} transmitted, over the air occupancy time T_{Total} due to these transmissions: $R_{avg} = B_{Total}/T_{Total}$. To claim that AP_W is facing a slow talker problem, we hypothetically speed up every other transmission observed in T_W. Let there be m other transmissions, with transmission i of size B_i and rate R_i. Compute the hypothetical data rate of transmission i as $H_i = max(R_i, R_{avg})$. Then we compute the air occupancy when there is no slow talker (i.e., all other transmissions at least as fast as R_{avg}) as $T_{hyp} = \sum_i B_i/H_i$. Whereas the original air occupancy is $T_{orig} = \sum_i B_i/R_i$. Now, S_{redn} is nothing but the additional fraction of airtime which would have been freed up had there been no one slower than AP_W: $S_{redn} = (T_{orig} - T_{hyp})/T_W$.

As computed above, we realize that S_{redn} is nothing but a sub-portion of the airtime represented by OBD and Ul. This is depicted in Fig. 3. As we shall show in our experiments, this computation of S_{redn} distinguishes mere presence of a slow radio (not sending much traffic) versus a slow radio causing throughput reduction for AP_W.

Number of contending transmitters (N)**:** The above set of airtime reduction metrics is sufficient to diagnose all of the problems grouped under "B. Not enough airtime" in Fig. 2, with the exception of the over-crowding problem. To diagnose over-crowding, we need a measure of the number of contending transmitters (denoted N), in T_W. Since T_W is typically a few seconds, which is much larger than typical frame transmission durations, we first break-up T_W into smaller windows t_s. The value of t_s must be large enough to observe multiple nodes' transmissions. We have seen that a t_s value of 100-500ms works well in practice (we use $t_s = 200ms$ in implementation). Within each t_s window, AP_W counts the number of observed transmitting radios, and takes this as an estimate of N. To compute N for a given T_W, we simply average these estimates over all the t_s windows within T_W.

Our experiments confirm that the above mechanism to estimate N is robust in diagnosing over-crowding. Finally, we note that the metric N is the only exception in our set of witals metrics in that it does not directly represent percentage throughput reduction due to a wireless problem.

B. Metrics for inefficient airtime utilization

We use three different metrics to diagnose the different causes under "C. Inefficient utilization of provided airtime" in Fig. 2. Each of these metrics represents throughput reduction in the given airtime.

Let us first calculate ideal throughput at the link layer. It is a function of link layer bit-rate R and the level of aggregation $Aggr$ (in case of 802.11n) [18]. It is of the form:

$$f(R, Aggr) = \frac{P \times Aggr}{P \times Aggr/R + T_{overheads}} \qquad (1)$$

Here, P is the payload size prior to aggregation, and $T_{overheads}$ accounts for overheads: the average DCF backoff, the PHY and MAC headers, 802.11 ACK, and TCP ACK. We capture the throughput reduction due to the three reasons *LowRSSI*, *RA*, and *LowAggr*, using $f(R, Aggr)$. The three witals metrics are denoted as $RSSI_{redn}$, RA_{redn}, and $Aggr_{redn}$ respectively.

These three metrics can be visualized as part of the same unified throughput reduction picture, as depicted in Fig. 4. A note of caution here is that the axes in the figure are non-linear; they correspond to $f(R, Aggr)$.

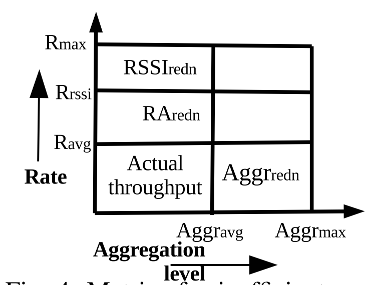


Fig. 4: Metrics for inefficient use of airtime

(a) $RSSI_{redn}$ represents the throughput reduction due to lower-than-ideal RSSI alone. It compares the throughput achievable at the highest data rate R_{max}, with that achievable at data rate possible ideally at the current RSSI, R_{rssi}. This comparison is done at the observed average level of aggregation $Aggr_{avg}$.

(b) RA_{redn}: Now, AP_W could be transmitting at a rate lower than R_{rssi}. Denote this as R_{avg}. RA_{redn} captures the throughput reduction from operating at R_{avg} as opposed to operating at R_{rssi}, at $Aggr_{avg}$.

(c) $Aggr_{redn}$ represents the throughput reduction at the observed rate R_{avg}, due to operating at the observed aggregation level $Aggr_{avg}$ as opposed to operating at the ideal $Aggr_{max}$.

Given the above, we can write, for each client i:

$$RSSI_{redn}(i) = \frac{f(R_{max}(i), Aggr_{avg}(i)) - f(R_{rssi}(i), Aggr_{avg}(i))}{f(R_{max}(i), Aggr_{avg}(i))}$$

$$RA_{redn}(i) = \frac{f(R_{rssi}(i), Aggr_{avg}(i)) - f(R_{avg}(i), Aggr_{avg}(i))}{f(R_{max}(i), Aggr_{avg}(i))}$$

$$Aggr_{redn}(i) = \frac{f(R_{avg}(i), Aggr_{max}) - f(R_{avg}(i), Aggr_{avg}(i))}{f(R_{avg}(i), Aggr_{max})}$$

We then compute each metric for AP_W as the average over the corresponding metrics for each active client i. We use an empirical definition of an active client: AP_W should have transmitted a minimum of 10 data packets to that client in T_W. For the above computations, we need $R_{max}(i)$, $R_{rssi}(i)$, and $Aggr_{max}$; the others are measured by AP_W over each T_W easily.

Here $R_{max}(i)$ is the ideal bit-rate for client i at the ideal RSSI; this is nothing but the maximum bit-rate: 54Mbps for 802.11g, 72 Mbps for 802.11n 1×1 etc.

To estimate $R_{rssi}(i)$ for client i, we use a combination of (a) the data rate determined by the measured average RSSI on the uplink, and (b) observed loss rates at the various data-rates in the past five T_W windows. This estimation of R_{rssi} is similar to computing SNR threshold for adapting bit-rate [24].

We have taken $Aggr_{max} = 10$, as this represents (an almost) ideal level of aggregation; as [18] shows, an aggregation level of 10 achieves close to optimal efficiency (88% for a bit-rate of 72Mbps).

V. Diagnosis Algorithm

We now design the diagnosis algorithm, based upon the causal graph of Sec. III, and the witals metrics of Sec. IV. The algorithm runs at the end of every T_W, using as input the witals metrics measured in that T_W. It gives as output one or more of the diagnoses listed in the first column of Table III. In our implementation, we choose $T_W = 2s$.

A. Algorithm

The algorithm is summarized in Algorithm 1. The tight connection between the witals metrics and the various nodes in the causal diagnosis graph, as captured in Table III, enables the design of the algorithm.

The algorithm first computes $\mathbb{M}$, representing the set of major throughput reduction metrics. Our design principle in witals, of making the metrics comparable is what enables the definition of $\mathbb{M}$. Among the metrics W, O, OBO, $OBD\ Ul$, S_{redn}, $RSSI_{redn}$, RA_{redn}, $Aggr_{redn}$ (i.e., all witals except N), we select the top three as representing the major reasons for throughput reduction. To account for the fact that even the top three among these may not be causing throughput reduction, we apply a *min-thpt-thresh* i.e., minimum throughput threshold of 10% to each of these metrics; i.e., $\mathbb{M}$ consists of up to three metrics, each of which represents at least a 10% throughput reduction.

We first diagnose *Healthy* if $\mathbb{M}$ is empty, i.e., there is nothing causing over *min-thpt-thresh* throughput reduction. Given a non-empty $\mathbb{M}$, the diagnosis of *CtlMgtOvhd*, *OtherBSSOvhd*, *OtherBSSData*, *Upload*, *SlowTalker*, *LowRSSI*, *RA*, *LowAggr*

Algorithm 1 Diagnosis algo, runs every T_W

1: $\mathbb{M}$ = Set of up to 3 major witals metrics $>$ *min-thpt-thresh*
2: **if** $\mathbb{M}$ is $\emptyset$ **output** Healthy
3: **if** $I >$ *idle-thresh* **then** ▷ Channel more idle than busy
4:     **output** LowLoad, Healthy
5: **else** ▷ Channel more busy than idle
6:     **if** $O \in \mathbb{M}$ **output** CtlMgtOvhd
7:     **if** $OBO \in \mathbb{M}$ **output** OtherBSSOvhd
8:     **if** $OBD \in \mathbb{M}$ **output** OtherBSSData
9:     **if** $Ul \in \mathbb{M}$ **output** Upload
10:     **if** $S_{redn} \in \mathbb{M}$ **output** SlowTalker
11:     **if** $RSSI_{redn} \in \mathbb{M}$ **output** LowRSSI
12:     **if** $RA_{redn} \in \mathbb{M}$ **output** RA
13:     **if** $Aggr_{redn} \in \mathbb{M}$ **output** LowAggr
14:     **if** $W \in \mathbb{M}$ **then**
15:         **if** $N >$ *simul-tx-thresh* **output** Overcrowding
16:         **if** $(OBO + OBD) >$ *otherbss-pres-thresh* **output** CoChInterf
17:         **if** neither of the above **output** NonWiFi
18:     **end if**
19: **end if**

is straightforward: we simply check if the corresponding metric (as given in Table III) is in $\mathbb{M}$ (lines 6-13 in Diagnosis Algo). We note that these "problems" are diagnosed only when the channel is more busy than idle (i.e., $I < 50\%$).

Now, from the causal graph (Fig. 2), we see that W (wasted non-decodable energy) could be caused due to a variety of reasons: non-WiFi device, or co channel interference, or over-crowding. Lines 14-17 of diagnosis algo distinguish between these possible causes. The metric N which is a measure of simultaneous transmitters on the channel is used to flag *Overcrowding*. Note that, we are measuring number of transmitters detected at the AP at small intervals, thus the actual number of simultaneous transmitters might be more than what the the AP sees. The transmitters who have have been successful to send their frames, are the ones seen by AP. Hence, we pick an empirical threshold of 5 for *simul-tx-thresh* (simultaneous transmitters threshold) for detecting overcrowding. The presence of significant interference from other BSS (*otherbss-pres-thresh*) is used to flag *CoChInterf*. If neither of the above is identified as a possible cause, we conclude that the high W is due to a *NonWiFi* device.

In several real-life measurements, we found that a diagnosis every T_W might be too fine-grained for a sysad, and it flags several transient problems. The sysad might be interested to know the persistent performance problems. Therefore, Witals also provides a further refined view showing the persistent performance problems in the last 15 minutes or so.

B. An Example Run

Now we take an example to better understand the diagnosis algorithm. We take a sample trace from a real-life measurement where 80 clients were associated with AP_W, the same as mentioned in Sec. I. The clients were tablets used by students in a classroom, and the traffic consisted of an online quiz.

Fig 1(a) plots the witals metrics over time. We note here that the state-of-the-art in current commercial systems stops with such a view of a multitude of graphs. Now, recall that each witals metric shows airtime or throughput reduction percentage because of that problem. Even with the metrics being comparable, Fig. 1(a) is not easy to interpret visually.

Fig. 1(b) next shows the output of our diagnosis algorithm plotted over time. The width of the bubble is proportional to the throughput reduction it causes. We also plot aggregate throughput of the network. We can see here that *Overcrowding*, *RA*, *OtherBSSOvhd*, *OtherBSSData* and *CoChInterf* are the major performance problems for this network. The sysad can potentially set event triggers based on the algorithm's output and take appropriate actions. Note that the distilled view in Fig. 1(b), is enabled by the design of *comparable* witals metrics and the cross-links. Such distillation is *not* possible in current commercial systems or state of the art prior work.

C. Parameters in the algorithm

In our Witals algorithm, we have used a few parameters. They are: time window of measurement $T_W = 2sec$, *simul-tx-thresh* of 5 for determining *Overcrowding*, slot size $t_s = 200ms$ for determining number of contending transmitters (N), *min-thpt-thresh* of 10%, *idle-thresh* of 50%, active clients' number of data packets threshold of 10, picking top 3 reasons, as opposed to some other number. The values of these parameters are not crucial to our diagnosis algorithm. In Sec. VI, we evaluate the sensitivity of the parameters.

VI. Evaluation

We evaluate our diagnosis algorithm in two different settings (1) controlled experiments and (2) real-life measurements. In controlled experiments, we artificially create a problem and see if the Witals diagnosis matches the ground truth. Next, we run our algorithm in real settings where we corroborate ground truth from system administrators and from users' feedbacks. In all cases, Witals matches ground truth accurately. Here, we also evaluate Antler [20] and WISE [19] prior work on AP-centric diagnosis (described in Sec II). We also compare Witals with popular commercial tools. Prior to the experiments, we first describe our prototype implementation.

A. Prototype Implementation

We have implemented Witals on an enterprise grade commercial AP[1] platform capable of 2×2 as well as 3×3 802.11n operation. The hardware is based on the Atheros 9xxx generation chipset, and the software is a set of modifications to a licensed version of the Atheros driver. The radio is placed in promiscuous mode, so that the driver can extract statistics for packets intended for other radios in the vicinity too. We have implemented the diagnosis logic as part of AP software suite. To measure the accounting of airtime at AP_W, we take help of hardware supported registers. We also log extensive info (address, length, data rate etc.) about transmitted and received frames. We had to pay attention to detail, in making the implementation work at the high rates of 802.11n. The simple set of witals metrics helped too, in keeping the overhead minimal: while some of the witals metrics such as S_{redn} are subtle in design, the per-frame computation itself is minimal. We studied the impact of Witals on AP_W's throughput, comparing it with a vanilla AP as well as the expected throughput as per Eqn. 1. We found that the throughput of the Witals AP closely matched the expected throughput, even at 2x2 802.11n 40MHz channel bonded operation.

[1] AirTight AP: AirTight C-55 AP, AirTight C-60 AP. http://www.airtightnetworks.com/home/resources/datasheets.html

We also implemented Antler [20] and WISE [19]. Antler uses many thresholds for diagnosis, but provides value for only two. So, we have assumed the missing threshold values to the best we can. We did not implement the Airshark module in WISE: this module differentiates between various sources of non-WiFi activities, and is complementary to our work. WISE performs an offline analysis using the $Witt$ metric. For comparison, we assume that the causes WISE attributes to poor performance are various diagnoses of WISE.

B. Parameter sensitivity

To evaluate the sensitivity of our chosen thresholds on Witals diagnosis, we change all the threshold values evaluate in the same classroom scenario (in Sec. V). We modify T_W as 5 sec, *simul-tx-thresh* as 10, t_s as $100ms$, *min-thpt-thresh* as 20%, *idle-thresh* as 60%, active clients' number of data packets threshold as 20. In all cases, our diagnosis does not change significantly (not shown here) with the changed values. Likewise, the sysad can specify 3/more/lesser major reasons for Witals diagnosis algorithm depending on the specific deployment demand. Therefore, our algorithm is not crucial to these thresholds. One can set an appropriate threshold that suits the deployment demand.

C. Controlled experiments

In the controlled experiments below, we use an 802.11n client associated with an 802.11n Witals AP (AP_W). We used iperf TCP throughput tests on the downlink; the iperf throughput was used to learn ground truth regarding presence of a "problem". In each controlled experiment, to the extent possible, we ensured the "problem" we intentionally introduced was the only problem. For each experiment, we plot the output of Witals and it matches the ground truth cause.

*1) **Detection of LowRSSI**:* We start with the simple scenario of poor signal strength detection: *LowRSSI*. For this experiment, AP_W's client was moved across four locations while staying for 30 sec at each location. We started at time $t = 11s$ at position 1 (1m from AP_W), moved to position 2 (5m) at $t = 60sec$, moved to position 3 (20m) at $t = 99s$, then moved to position 4 (40m) at $t = 142s$, then finally came back to position 1 at $t = 212s$.

Initially, Witals diagnoses (plot not shown) the network as *Healthy*. But, during the time the client faced poor signal strength (between 100-200s, at position 3 & 4), *LowRSSI* is diagnosed. We also observe that the bit-rate was sub-optimal in many T_W windows, likely due to the rate adaptation algorithm exploring the best rate in the face of client mobility. There are also several instances of *LowAggr* diagnosed: as throughput reduces, this appears to interact poorly with the A-MPDU aggregation algorithm as well. Using a plot of the raw witals metrics against time (not shown), we found that $RSSI_{redn}$ was around 30% in position-3 and close to 100% in position-4.

Comparison with Antler and WISE: We evaluated Antler and WISE against the trace from this experiment. Both are able to detect poor link condition.

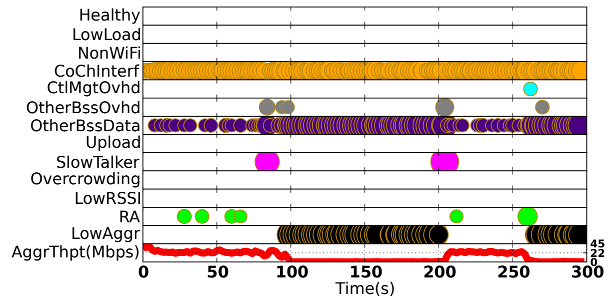


Fig. 5: Witals diagnosis: *OtherBSS*

*2) **Detection of NonWiFi device presence**:* To test Witals on detection of presence of non-WiFi device: *NonWiFi*, we set up an experiment where we operate AP_W in channel-9 of the 2.4GHz band. We used a microwave oven, specified to operate around the same central frequency, as the interferer. From a plot of Witals diagnosis (not shown), we find that *NonWiFi* is diagnosed in this time window. Also diagnosed frequently is *RA*, as the microwave causes high frame loss, and AP_W's rate adaptation algorithm confuses this for channel loss.

During the time the microwave was on, the aggregate throughput dropped to about 10Mbps from about 23Mbps. A plot of the witals metrics (not shown) indicated that W (wasted non-decodable energy) was causing a 45% reduction in airtime and throughput.

Comparison with Antler and WISE: While WISE is able to detect the presence of non-WiFi device, Antler provides a diagnosis of interference. Antler reports interference if there is a difference between the AP's airtime and the airtime attributable to its clients' uplink and downlink transmissions. Such a difference arises in this experiment due to a high degree of wasted non-decodable energy W resulting from microwave. However Antler is unable to detect the reason for the interference i.e., non-WiFi device.

*3) **Detection of OtherBSSData**:* In this experiment, we introduced interference from another BSS using a second AP (say AP2) operating on the same channel as AP_W and kept close to AP_W. We also had a client (C2) associated with and kept close to AP2, with iperf-generated downlink TCP traffic. Fig 5 shows the result of Witals algorithm. We see that presence of high data traffic on the other BSS is correctly diagnosed, and so is the co-channel interference (*CoChInterf*) caused by AP2. During several T_W windows, *LowAggr* is also reported, like in the *LowRSSI* experiment.

The ground truth was further corroborated by looking at a plot of the witals metrics over the same time window (not shown). We found that the OBD metric was around 20% until about $t = 100s$. Subsequently, AP2 captures a larger share of airtime, OBD increases to 70%.

Comparison with Antler and WISE: Here, Antler diagnoses interference, and WISE detects high airtime utilization from external sources. Antler is unable to classify this interference as : *NonWiFi*, *CoChInterf*, *OtherBSSOvhd*, or *OtherBSSData*. WISE too can not classify the reason for high utilization as : *CoChInterf*, *OtherBSSOvhd*, or *OtherBSSData*.

*4) **Detection of SlowTalker**:* To create a situation of MAC unfairness due to a slow radio, we set up an experiment where three clients are associated with AP_W. Two of them were running iperf in the downlink direction and third one

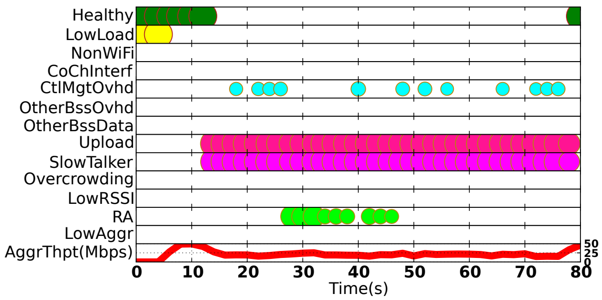


Fig. 6: Witals diagnosis: *SlowTalker*

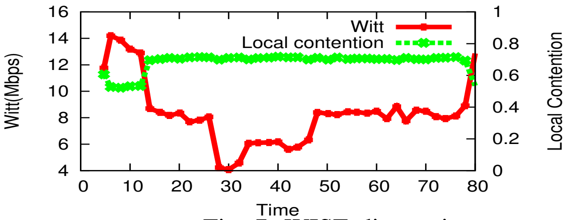


Fig. 7: WISE diagnosis

was running iperf in the uplink direction. Here we fixed the uplink client's transmission bit-rate to be low: MCS 0, 6.5Mbps/7.2Mbps. Fig 6 shows the diagnosis output of Witals algorithm against time. We see that *SlowTalker* is correctly diagnosed as a major reason for throughput reduction. We also see that *Upload* is a significant reason. A plot of the witals metrics (not shown) indicated that the slow client occupies almost 60% of the channel (this shows up as *Ul*) and AP_W's downlink airtime is only around 26%. The slow talker reduction metric (S_{redn}) is around 60%.

Comparison with Antler and WISE: Here, Antler can not diagnose the slow talker problem. WISE provides wrong diagnosis of high local contention (> 0.5 [19]). WISE computes Witt metric using $link_{exp}$ model, which in turn computes the local contention. Fig 7 shows the Witt value & local contention with time. The reason for wrong diagnosis is that WISE diagnoses local contention in an indirect way, from the relative amount of other clients' traffic to the total traffic. Since the slow client occupied the channel for more time, other clients could not get enough airtime. Hence, WISE diagnoses this as high local contention for the fast clients.

*5) **Detection of Overcrowding**:* To test our Witals framework's ability to detect *Overcrowding*, we set up an experiment where 30 clients were associated with AP_W. Before the experiment, we ensured the access point placement was optimal so that all clients can operate at the highest bit-rate individually. We also ensured that the web server or the wired network were not loaded. That is, the performance seen by the clients in our experiment was constrained by the wireless network bottleneck. External WiFi interference was minimal. All the clients were downloading a large file from a local server and simultaneously uploading another large file.

Even though a controlled experiment, this scenario stress tests our Witals framework as several phenomena kick-in simultaneously. As all the clients were downloading as well as uploading, the number of contending stations is high. This causes collisions, which in turn results in airtime wasted in non-decodable energy (cross-links 3 & 4, Fig. 2). Collisions also cause frame losses, which in turn cause the rate adaptation algorithm to select a sub-optimal bit-rate (cross-link 6, Fig. 2). Retransmissions & variable RTT could also cause TCP slowdown (cross-link 5, Fig. 2).

Despite the above simultaneous phenomena, we see in Fig 8(a) that Witals algorithm presents a clear picture. It diagnoses *Overcrowding* correctly. We see that the *SlowTalker*, *CtlMgtOvhd*, and *RA* problems are reported as well. A plot of the witals metrics over time (not shown) indicated that W varied between 50% and 80% during the experiment. The metric O was also high, about 30%, likely due to collision induced FCS errors. We observed that the combination of the above factors reduced the downlink throughput to near zero during the experiment.

Comparison with Antler and WISE: Antler infers congestion from bit-rate. If bit-rate is low but RSSI is high, it diagnoses as congestion. We evaluated Antler against the trace from this experiment. Fig 8(b) shows the diagnosis. Instead of reporting congestion, the diagnosis was *interference*. The incorrect diagnosis is due to difference between the AP's airtime and the airtime attributable to its clients' uplink and downlink transmissions. High degree of wasted non-decodable energy W resulting from collisions causes this difference.

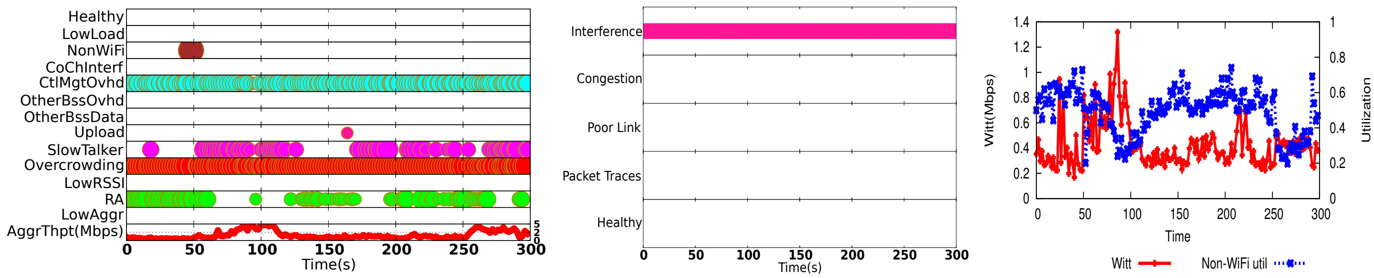
Here, WISE too detects interference. Fig 8(c) shows the Witt value & non-WiFi energy. The reason for high non-WiFi energy is high degree of wasted non-decodable energy W resulting from collisions. WISE diagnoses this as interference. Our diagnosis graph specifically cross-links 3, 4 of Fig. 2, and their incorporation into our diagnosis algorithm allows us to flag the correct diagnosis.

*6) **Witals on VoIP performance**:* So far, we have seen that Witals is able to detect the "low throughput" issues correctly. To understand, whether Witals framework can detect the issues with poor VoIP performance, we set up a controlled experiment. Initially, one client connects to AP_W. We emulate VoIP traffic via UDP CBR (constant bit-rate) traffic of 1.2 Mbps (Skype HD video [4]). After 80 sec, we connect another client to AP_W. This client runs iperf in the uplink direction for another 100 sec. Fig 9 shows the diagnosis output of Witals algorithm against time. Here, instead of aggregate throughput, we plot jitter values with time. We see that initially the jitter values are low, but once the upload starts the jitter values increase. Witals is also able to detect the reason for increase in jitter as *Upload*.

D. Real life measurements

We tested Witals in various real scenarios. Here, we discuss results from 4 different real scenarios: (1) a research conference with 10 clients connected to the AP under diagnosis, (2) a classroom at our Institute with 43 clients connected to AP_W (termed classroom-43 henceforth), (3) a denser classroom at our Institute with 80 clients to AP_W (termed classroom-80 henceforth), and (4) a students' hostel (dormitory) in our Institute, with 20 clients connected to the AP under diagnosis. In each case, we corroborated ground truth in terms of presence of wireless problem(s) by verifying with system administrator(s), collecting users' feedback, and also our own observation. We also verified ground truth by manually looking into our traces.

In the classroom settings above (classroom-43 and classroom-80), AP_W was the one used for network access by



(a) Witals diagnosis (b) Antler diagnosis (c) WISE diagnosis

Fig. 8: *Overcrowding* / *Congestion* Detection Witals Vs Antler & WISE

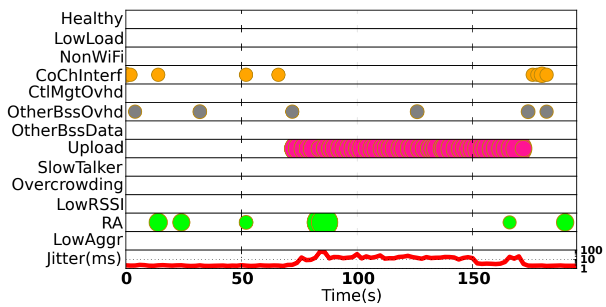


Fig. 9: Witals diagnosis for VoIP application

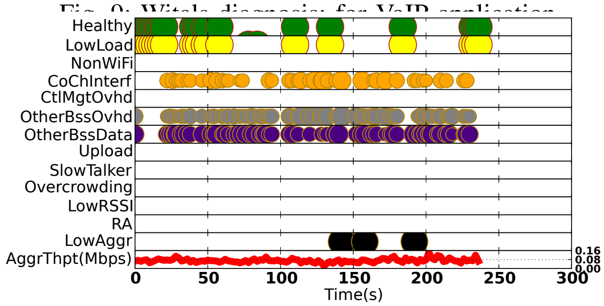


Fig. 10: Witals diagnosis: Conference

the clients. In the conference and hostel settings however, none of the clients associated with the Witals AP AP_W. The Witals radio AP_W collected traces and metrics all the same, acting like a sniffer on the same channel: we tweaked the algorithm implementation to perform the diagnosis at AP_W on behalf of an actual operational AP to which clients connect.

*1) **Conference**:* We evaluated Witals at a research conference. In this setting, there were several neighboring APs in various 2.4GHz channels. Although, only about 10 clients were connected to the AP, the performance of WiFi was really poor. Users complained about slow WiFi. Fig. 10 shows the diagnosis output of Witals algorithm for a representative 5-minute duration. There are several time windows where the WiFi network is lightly loaded or otherwise healthy (user listening to an interesting talk?). But the major performance problems in this network are *OtherBSSData*, *OtherBSSOvhd* and *CoChInterf*. *OtherBSSOvhd* is an unexpected diagnosis. In this setting, there were several APs that operate on the same channel. So, one can understand the data frames of other BSS causing performance problems. Again, some of the energy that is non-decodable at AP_W is causing co-channel interference. The remedy action for these two problems could be suggesting the clients of other APs to move to another AP on different channel and likewise. But control, management overheads of other BSS i.e., *OtherBSSOvhd* will still be there. This implies poor configuration of the APs.

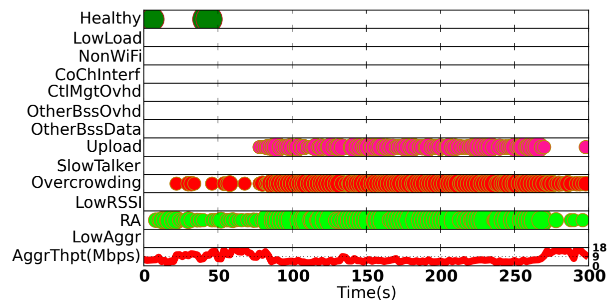


Fig. 11: Witals diagnosis: classroom-43

Validation: We conveyed our Witals diagnosis to the concerned system administrators. They were surprised to know the diagnosis and identified the issues in their configuration.

*2) **Classroom-43**:* In this classroom setting, students downloaded online video instructional material over WiFi, during lab hours. The setup consisted of students connecting to a web server that hosts the video content. The students used WiFi-enabled individual laptops and tablets, and some used wired desktops as well. The activity was as follows: the students connected to content server, authenticated themselves, and viewed embedded video content (with javascript controlled video markers) over the network. Fig. 11 shows the diagnosis output of Witals algorithm for a representative 5-minute duration. It shows the aggregate throughput to be about 6 Mbps, significantly lower than ideal. We see that there is significant presence of *Overcrowding*, which matches with the empirical ground truth observation. The diagnosis also highlights two other significant problems: *Upload* and *RA*. *Upload* was a surprising diagnosis, since most students were simply expected to watch the downloaded videos. And the *RA* problem is a side-effect of the over-crowding.

Validation: For validation of our diagnosis we collected users' feedbacks. The authors were also present in the classroom. The empirical observation was that the network was noticeably slow when all students were using the network. Students complained to the teacher that their downloads were getting stuck. In the anonymous feedbacks, students revealed that they were using facebook, whatsapp etc. in classroom. That validates our surprising diagnosis of high upload. Prior work [17] also reported similar problems in a dense classroom.

*3) **Classroom-80**:* We also tested Witals in another highly crowded WiFi-enabled classroom at our Institute. Here almost 200 students were present. We set up 3 access points in 3 non overlapping channels of 2.4Ghz ISM band. There were many neighbouring APs on each of the non overlapping channels.

The WiFi network was used for a classroom activity involving an online quiz using an Android quiz application on the tablets. The students logged in via the app to a server, when the quiz was published by the teacher, the app downloaded the quiz file. The app enabled students to answer the questions, after which it submitted the answers to the server.

The ground truth observed empirically was that the students' experience was really bad. The app took a long time to download the quiz file. When submitting responses also, many students found their upload was stuck. Out of 200 students only about 120 students could submit their responses, with the instructor having to fall back on paper-printed quiz forms. Given that quiz file itself was small, we did not expect such poor performance. The AP_W for which we present results served 80 clients. Fig. 1(b) (shown in Sec. I) plots Witals diagnosis for a representative 5-minute window.

We see here that once again *Overcrowding* is reported as a major reason. Unlike classroom-43, *Upload* is not a problem, while *OtherBSSData* and *OtherBSSOvhd* are significant problems. *Overcrowding* also results in *RA*, and *OtherBSSData* and *OtherBSSOvhd* also result in *CoChInterf*; these combined effects reduce throughput significantly, explaining the poor user experience. The value of this refined view can be understood by trying to deduce the same visually, from the raw witals metrics in Fig. 1(a) (in Sec. I).

Validation: For validation of our diagnosis we spoke to system administrator of our Institute. They confirmed that indeed there are many BSS on each of the non overlapping channel which was out side their control. The students also complained about the slow network. The authors were also present at the venue and they too experienced slow network.

4) ***Hostel****:* Finally, we present results of testing Witals at a student hostel (dormitory) in our Institute campus. The hostel access point serves around 100 student-rooms across 4 floors. We placed our AP_W close to the hostel's AP, to enable Witals diagnosis. The empirical ground truth here was that the user experience was satisfactory.

While Witals was operational here for a whole day, we look at a 5-min window during a busy period (around 2.50am!), as the network is mostly idle otherwise. Even in the relatively busy period, the network is lightly loaded and healthy for several T_W windows. Witals diagnoses (not shown here) *LowRSSI* and *CtlMgtOvhd* as significant reasons. Sub-optimal rate selection is a problem here too, as indicated by *RA*, but unlike the previous classroom scenarios, Witals algorithm does not attribute this to collision, but to other issues: it is likely that there are client-side reasons for such sub-optimal rates.

We also see in this measurement, the benefit of our definition of S_{redn}. There are several slow radios, but they are not sending sufficient traffic to cause a throughput problem and trigger a *SlowTalker* diagnosis in Witals. Had we used a simple ratio of fast to slow radio, like in [22], several instances of *SlowTalker* would have been incorrectly diagnosed.

Validation: For validation of of our diagnosis we spoke to system administrators of the hostel. They informed that one access point serves several floors and there is coverage issue in several rooms. This validates our diagnosis of *LowRSSI*. The problems reported above were short-term, the network was *Healthy* and *LowLoad* for quite a few time windows, thus it did not affect user experience: this matched with our observation that there were no major complaints by users.

E. Comparison of Witals with Commercial Tools

We compare Witals with popular commercial tools. These tools plot various metrics incomparable to each other and uses arbitrary thresholds. Further, they do not consider various simultaneous cause-effects of wireless problems. We come up with some example scenarios and compare Witals diagnosis with popular commercial tools such as AirDefense [5].

In presence of other BSS (in otherBSS controlled experiment Sec VI-C), AirDefense will correctly trigger an interference alarm. Here, Witals also correctly diagnoses *OtherBSSData* and related *CoChInterf*. AirDefense will trigger an utilization alarm when control/management/data frames or number of associations exceed the thresholds. For validation, we conducted a controlled experiment where 90 clients (tablets) were associated with our Witals-enabled AP for a classroom quiz. The load on the network was very low, aggregate throughput is merely 100Kbps. Witals diagnoses the network as *LowLoad* and *Healthy*. But in this case, AirDefense will trigger an utilization alarm. Most commercial tools diagnose any non 802.11 energy as interference. But during high collisions, when two frames collide, the energy becomes non-decodable. These tools can not differentiate non-decodable energy caused by external sources from the one caused by high collisions. For validation, we take the overcrowding controlled experiment described in Sec VI-C. Fig 8(a) shows that Witals correctly diagnoses *Overcrowding*, but most commercial tools will diagnose this as *Interference*.

VII. Discussion and Future Work

We now discuss a few aspects related to Witals.

What about 802.11ac? While we have built a prototype of Witals for 802.11a/b/g/n, fundamentally the algorithm remains same for 802.11ac as well. We are in process of porting our Witals implementation to a 802.11ac driver.

What about uplink? Conceptually, an instance of Witals can run at the clients and will be able to diagnose uplink performance issues. But due to huge variability of clients' drivers the implementation might be difficult. On the other hand, if we diagnose at the access point we will be losing out some important information. For example, when the network is lightly loaded, AP can not distinguish whether: clients are simply not generating enough traffic, or there is a wireless issue. There are several client side diagnosis tools such as WiFi-Profiler [9], Client-Conduit [7]. These are complementary to our work, we can install these tools at the clients and collect client side information, which could be potentially utilized in the Witals framework to diagnose uplink performance problems.

What about Multiple APs? Witals can be potentially extended to understand the performance of broader deployments for enterprise/campus networks by co-ordinating information across multiple vantage points. This requires a coarse grained synchronization (in granularity of T_W) between the APs and a central server for analysis. Each AP can send Witals diagnosis to a central server. Then the server has to merge each AP's diagnosis to provide a combined view of the network. The diagnosis of *LowLoad*, *NonWiFi*, *Overcrowding*, *LowRSSI*, *RA*, *LowAggr* will remain same. The server will note the frequency of such reportings in all the APs and will flag-off the majorities. Now, *OtherBSSOvhd* and *OtheBSSData* will no longer be diagnosed from a channel perspective. Instead, these will be *CtlMgtOvhd* and *Download*.

Before-and-after analysis? We have performed several controlled experiments to understand how Witals helps to diagnose issues and improve performance. For example, in one controlled experiment, 15 laptops simultaneously downloaded and uploaded a long file. Here, Witals reported *Overcrowding* and *RA*. After diagnosing overcrowding as the root cause, we plugged in a solution WiFiRR [17] to reduce congestion. Thereafter, Witals reported *Healthy* and *LowLoad*. The throughput also improved from from 10-12 to 20-40 Mbps. The details are described in [16].

VIII. Conclusion

In this paper, we have presented the design of Witals, an AP-centric system for diagnosing a WiFi network's health. We do not require any distributed sensor infrastructure, extra radio at the AP, or client-side modifications. We start with the design of a causality graph. The causal graph identifies an extensive set of performance problems. We then come up with a minimal set of metrics, witals, closely associated with the nodes of the causal graph. An important property of the witals metrics is that each of them quantifies throughput reduction, and being in the same units, are comparable. Our diagnosis algorithm then puts together the causal graph and the metrics. The final output is capable of presenting a clear picture to a sysad, even in the presence of complex real-life WiFi network problems. The Witals framework is capable of detecting multiple simultaneous problems too, a common occurrence. We have evaluated Witals extensively in controlled as well as real-life uncontrolled settings, based on a prototype. We show that our diagnosis matches ground truth closely. Witals also helped the sysads unearth some unexpected root-causes behind performance degradation in certain deployments. We also perform a quantitative comparison of Witals with research and commercial tools.

Given the importance of WiFi performance diagnosis, especially in cloud-managed remote WiFi deployments, we believe that the AP-centric Witals framework is a significant contribution, even as WiFi speeds grow.

IX. Acknowledgement

This work was supported in part by Mojo Networks.

References

[1] "7 Signal", http://7signal.com/solutions/wi-fi-performance-solution/.
[2] "AirMagnet WiFi Analyzer", http://www.flukenetworks.com/enterprise-network/wireless-network/AirMagnet-WiFi-Analyzer.
[3] "Cisco Meraki" https://meraki.cisco.com/solutions/high-density-wifi.
[4] "How much bandwidth does Skype need?". https://support.skype.com/en/faq/FA1417/how-much-bandwidth-does-skype-need.
[5] "Motorola Airdefense". www.airdefense.net.
[6] "WiFiEagle", http://nutsaboutnets.com/wifieagle-channel-analyzer/.
[7] A. Adya, P. Bahl, R. Chandra, and L. Qiu. Architecture and Techniques for Diagnosing Faults in IEEE 802.11 Infrastructure Networks. In *MobiCom*, 2004.
[8] K. V. Cardoso and J. F. de Rezende. Increasing throughput in dense 802.11 networks by automatic rate adaptation improvement. *Wireless Networks*, 2012.
[9] R. Chandra, V. N. Padmanabhan, and M. Zhang. WiFiProfiler: Cooperative Diagnosis in Wireless LANs . In *MobiSys*, 2006.
[10] Y.-C. Cheng, M. Afanasyev, P. Verkaik, P. Benko, J. Chiang, A. C Snoeren, S. Savage, and G. M. Voelker. Automating Cross-Layer Diagnosis of Enterprise Wireless Networks. In *SIGCOMM*, 2007.
[11] Y.-C. Cheng, J. Bellardo, P. Benko, A. C. Snoeren, G. M. Voelker, and S. Savage. Jigsaw: Solving the Puzzle of Enterprise 802.11 Analysis. In *SIGCOMM*, 2006.
[12] J. Kim, S. Kim, S. Choi, and D. Qiao. CARA: Collision-Aware Rate Adaptation for IEEE 802.11 WLANs. In *INFOCOM*, 2006.
[13] K. Lakshminarayanan, S. Seshan, and P. Steenkiste. Understanding 802.11 Performance in Heterogeneous Environments. In *HomeNets*, 2011.
[14] S. C. Liew, C. H. Kai, H. Ching Leung, and P. Wong. Back-of-the-Envelope Computation of Throughput Distributions in CSMA Wireless Networks. *IEEE Transactions on Mobile Computing, 9(9),*, 2010.
[15] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Analyzing the MAC-level Behavior of Wireless Networks in the Wild. In *SIGCOMM*, 2006.
[16] M. Maity. *Health Diagnosis and Congestion Mitigation of WiFi Networks*. PhD thesis, Indian Institute of Technology, 2016.
[17] M. Maity, B. Raman, and M. Vutukuru. TCP Download Performance in Dense WiFi Scenarios: Analysis and Solution. *IEEE Transactions on Mobile Computing*, 2017.
[18] N. Mishra, A. Chaurasia, A. Kallavi, B. Raman, and P. Kulkarni. Usage of 802.11n in Practice: A Measurement Study. In *COMSNETS*, 2015.
[19] A. Patro, S. Govindan, and S. Banerjee. Observing Home Wireless Experience through WiFi APs. In *MobiCom*, 2013.
[20] R. Raghavendra, P. A. K. Acharya, E. M. Belding, and K. C. Almeroth. Antler: A multi-tiered approach to automated wireless network management. In *INFOCOM Workshops 2008, IEEE*, 2008.
[21] A. Sheth, C. Doerr, D. Grunwald, R. Han, and D. Sicker. MOJO: A distributed physical layer anomaly detection system for 802.11 WLANs. In *MobiSys*, 2006.
[22] V. Shrivastava, S. Rayanchu, S. Banerjee, and K. Papagiannaki. PIE in the Sky: Online Passive Interference Estimation for Enterprise WLANs. In *NSDI*, 2011.
[23] S. Sundaresan, N. Feamster, R. Teixeira, Y. Grunenberger, D. Papagiannaki, D. Levin, et al. Characterizing Home Network Performance Problems.hal-00864852. 2013.
[24] J. Zhang, K. Tan, J. Zhao, H. Wu, and Y. Zhang. A practical SNR-guided rate adaptation. In *INFOCOM*, 2008.

Mukulika Maity Mukulika Maity received her B.E in CSE from BESU in 2010. She received her M.Tech + Ph.D. dual degree in CSE from IIT-Bombay in 2016. Since Dec 2016, she is a faculty at the CSE department at IIIT- Delhi. Her research interests are broadly in the areas of wireless networks and mobile computing.

Bhaskaran Raman Bhaskaran Raman received his M.S. and Ph.D. in CS from U.C.Berkeley, in 1999 and 2002 respectively. Since July 2007, he is a faculty at the CSE department at IIT-Bombay. His research interests and expertise are in communication networks, wireless/mobile networks, and technology for education. His current focus and specific interests are mobile crowdsensing, WiFi performance diagnosis, and use of technology in large classrooms for effective teaching.

Mythili Vutukuru Mythili Vutukuru received her Ph.D. and M.S. degrees in Computer Science from the Massachusetts Institute of Technology in 2010 and 2006 respectively. Since July 2013, she is a faculty at the CSE department at IIT-Bombay. Her research interests are in networked systems, wireless communication, and network security.

Avinash Chaurasia Avinash Chaurasia received his B.Tech in CSE from NIT Allahabad in 2008. He obtained his M.Tech degree in CSE from IIT Kanpur in 2011. Currently he is pursuing Ph.D from IIT Bombay and working on areas related to networks and systems.

Rachit Srivastava Rachit Srivastava received his B.Tech in ECE from MNNIT in 2009. He obtained his M.Tech degree in ECE from IISC Bangalore in 2011. Currently he is a lead Engineer in Mojo Networks.