SmartAssoc: Decentralized Access Point Selection Algorithm to Improve Throughput

Fengyuan Xu, *Member, IEEE,* Xiaojun Zhu, Chiu C. Tan, *Member, IEEE,*
Qun Li, *Member, IEEE,* Guanhua Yan, and Jie Wu, *Fellow, IEEE*

Abstract—As the first step of the communication procedure in 802.11, an unwise selection of the Access Point (AP) hurts one client's throughput. This performance downgrade is usually hard to be offset by other methods like efficient rate adaptations. In this article, we study this AP selection problem in a decentralized manner with the objective of maximizing the minimum throughput among all clients. We reveal through theoretical analysis that the selfish strategy, which commonly applies in decentralized systems, cannot effectively achieve this objective. Accordingly, we propose an online AP association strategy that not only achieves a minimum throughput (among all clients) that is provably close to the optimum, but also works effectively in practice with a reasonable computation and transmission overheads. The association protocol applying this strategy is implemented on the commercial hardware and compatible with legacy APs without any modification. We demonstrate its feasibility and performance through real experiments and intensive simulations.

1 Introduction

It is already very common for Wireless LAN clients like mobile phones and laptops to face multiple choices of APs because of the high density deployment. Which AP to attach is not a trivial question especially when there are a lot of nearby users who may interfere with each other. A user selecting an inappropriate AP will experience bad service, or even hurt other users' throughput. The current technique of AP selection is for the user to selfishly pick the AP with the strongest signal, or RSSI value. The intuition is that factors like multipath effect and path loss which reduce network throughput will have a smaller effect when the user is communicating with an AP with a larger RSSI.

This simple strategy might fail when there is a large number of users crowded together. Consider the case when we have two APs on orthogonal channels, one with much stronger signal strength than the other, and a collection of users. All the users will simply pick the same AP (with the largest RSSI), so that the actual throughput of each user is very small because of channel contention between users. Based on this observation, alternative criteria such as selecting the AP which yields the largest throughput have been suggested.

However, it is unclear how well this *selfish* strategy will perform when every user attempts to connect to the AP which is able to maximize their own throughput. Unlike an AP's RSSI value measured by a user which is not affected by additional users associating with that AP, the AP's throughput will change as more users join in. Therefore, a user selecting an AP based on throughput may have to switch APs constantly, hence lowering overall performance.

In practice, we believe a good performance is to achieve the maximized minimum throughput for all clients. Using this simple but reasonable metric, we seek to design a practical distributed protocol for AP association. We theoretically analyze the worst-case performance of the selfish strategy, and introduce an online algorithm that achieves a better worst-case performance. Incoming user employing this algorithm determines an irrevocable association, only making use of the load information on the nearby APs, in order to minimize the $\mathbb{L}_p$ norm of the loads on all APs at the moment. Based on our online algorithm, we have implemented an association protocol, SmartAssoc, for commodity hardware driver at the client side. This protocol works well with current legacy 802.11 APs. Using a combination of real experiments and extensive simulation, we demonstrate that our online association protocol performs better than the RSSI-based and selfish AP selection.

Our main contributions are:

- We have designed a distributed online algorithm for AP association. This algorithm well captures interference in transmission. It only needs to be performed once on a new user when she or he joins the network, and meanwhile all existed users do not have to revoke their association decision already made. We have proved our algorithm is $e \log m$ competitive, while no online algorithm is able to do better than $\lceil \log(m+1) \rceil$ [1].
- Our implementation of SmartAssoc is practical and

- *F. Xu and Q. Li. are with the Department of Computer Science, College of William and Mary, Williamsburg, VA 23187. E-mail: {fxu,liqun}cs.wm.edu .*
- *X. Zhu is with the Department of Computer Science and Technology, Nanjing University, Jiangsu 210093, China. E-mail: gxjzhu@gmail.com .*
- *C.C. Tan and J. Wu are with the Department of Computer and Information Sciences, Temple University Philadelphia, PA 19122. E-mail: {cctan,jiewu}@temple.edu .*
- *G. Yan is with the Information Sciences Group, Los Alamos National Laboratory, Los Alamos, NM 87545. E-mail: ghyan@lanl.gov .*

does not require any modification on APs, making our technique applicable to existing wireless networks. A light-weight method is introduced to estimate one user's throughput on the target AP without association, reducing the operation overhead. SmartAssoc demonstrates its practicability in real experiments, as well as in large scale simulation settings.

The rest of this paper is as follows. We give an overview of the related work in Section 2, and explain our motivation in Section 3. Section 4 proposes our online algorithm as the association strategy for the protocol design of SmartAssoc, and characterize its performance properties through theoretical analysis in a realistic model. We present the practical and efficient implementation of the proposed association protocol on the off-the-shelf wireless LAN adapter in Section 5. SmartAssoc is demonstrated through real experiments in Section 6, and simulation results are provided in Section 7. Finally, Section 8 concludes.

2 Related Work

AP association plays an important role in improving wireless performance [2]–[12]. The signal-noise ratios, which is popularly used for access point selection in 802.11, has demonstrated as a bad idea by G. Judd et al. [13]. In their paper, some other criteria like AP load information are taken into consideration for improving the network performance and achieving load balancing. Metrics evaluating the AP load are interested by both academic community and industry. [14] relies on passive measurement of delay intervals between the time when a beacon is scheduled for transmission and its eventual transmission to estimate an AP's load. In [15], a combined metric is applied, which includes the number of stations associated, mean RSSI for the station set of AP and regular RSSI. Similar AP-assisted approach is proposed in [16], to give associated clients the information about load through beacons. Compared with [12], we complete our theoretical analysis and conduct new real experiments in this paper.

The selfish behavior of users in a congestion game has been studied theoretically. A special case where each user's decision is a singleton set is considered in [17], while [18] describes a class of congestion game where the payoff function associated with each resource is user specific. The convergences under different load balancing scenarios are provided in [19]. In this work we model the decentralized AP selection with selfish users as an extension of the weighted singleton congestion game in which the weight of a user varies as the associated AP changes. Other modelings of the wireless infrastructure selection include [20] and [21], targeting at different scenarios and goals. The major distinction of our work is that we design and implement on the commercial chipset an online greedy AP selection protocol in the distributed manner. The performance is supported by theoretical proof and demonstrated both in real experiments and simulations.

Compared with decentralized methods, work by [22] and [23] uses the idea that better AP association decisions can be obtained by relying on a global view of the entire WLAN, or an extra centralized controller. AP side system is modified in [23] to aggregate workload information and provide association control according to it. In [22], a more complicated central scheme for AP association is discussed. However infrastructure change is required.

There are also other papers discussing how to multiplex multiple APs. In [24], it created multiple virtual interfaces based on one single wireless card, and made them communicate with associated APs simultaneously. The paper [25] built a multi-interface association mechanism to distribute a client's data traffic on multiple accessible APs in a scenario where the backhaul link is the bandwidth bottleneck.

We take a practical wireless model that well captures the complicated interference among APs and clients. Considering interference like [26] makes great impact on the real world performance of algorithms designed upon this model. From the technical perspective, the load balancing literature provides the foundation of our solution to the formulated problem of AP association. We borrow results from the literature [1], [27], [28] to better understand the hardness of our formulated problem. We elaborate each work later when it is used. Load balancing also finds applications in other fields, such as wireless sensor networks [29].

3 System model and Motivation

We consider an IEEE 802.11 infrastructure network [30], in which there are m APs and n stationary clients or users. Given no central controller and global information, all the clients are allowed to freely choose an AP within the transmission range to associate with. Under this situation our goal is *to maximize the minimum throughput over all clients*, taking both overall throughput and fairness into consideration. Through this paper, we consider the MAC layer throughput if no specification.

In current IEEE 802.11 network protocol, a client is willing to associate with the AP that gives the strongest signal. We term this association strategy as Best-RSSI. However, the Received Signal Strength Indication (RSSI) is not a good indicator of throughput changes that a user cares about. [12] demonstrates this poor correlation between RSSI and throughput in practice. Since RSSI failed to reflect the actual performance of an AP, it is highly possible for many clients nearby to connect to the same AP, leading to the congestion.

For the sake of comparison, we introduce a widely used evaluation concept, competitive ratio. Competitive ratio is the performance ratio, with respect to some metric, between worst outcome of certain association protocol and the optimal strategy case. Here we apply the competitive ratio in terms of minimum client

throughput. According to the 802.11 standard, all clients connecting to the same AP share the same throughput. Therefore the competitive ratio of minimum client throughput is also equal to the ratio of maximum AP load if we define an AP's load, $\mathcal{L}$, as the reciprocal of its client's throughput (Definition 1). In above standard protocol scenario, this ratio can be as bad as $1/m$. When m is large, this plummeting performance becomes unacceptable.

Naturally people may ask whether this problem can be solved if current standard protocol directly utilizes throughput information instead of RSSI? To answer this question, we formulate a weighted congestion game where all clients are selfish and try to associate with any AP offering best throughput for it in the whole network. The outcome of this game represents the best performance this alternative is able to reach in the ideal case. We prove that its competitive ratio is $\Omega(m)$ and the whole network needs long time to converge to this point because clients greedily keep revoking their association decisions for better throughput. For details please refer to the first section of supplementary file.

In the following section, we propose an online algorithm with competitive ratio $O(\log m)$, which is an exponential improvement.

4 Online Algorithm

In this section, we introduce our practical online association strategy. Our online algorithm considers the communication load including both interference and congestion, which provides a more realistic model.

For any user u and AP a, we use R_{ua} to denote the *transmission rate* under the situation only u is associating with a. R_{ua} varies even for the same user. For the rest of this section, unless otherwise specified, the transmission rate refers to the effective transmission rate, which considers the overhead caused by retransmissions, random backoff and so on. Then we introduce the load definition of an AP (Definition 1). According to Lemma 1 (proven in Section 2 of the supplementary file), maximizing the minimum throughput over all clients is equivalent to minimizing the maximum load over all APs.

Definition 1: The load of an AP a, $\mathcal{L}_a$, is

$$\mathcal{L}_a = \sum_{i \in U_a} \mathbb{L}_{ia} = \sum_{i \in U_a} \frac{1}{R_{ia}}$$

where $\mathbb{L}_{ua}$ is the reciprocal of the transmission rate of user u on AP a.

Lemma 1: All the users on an AP a have the same throughput T_a

$$T_a = \frac{1}{\sum_{i \in U_a} \frac{1}{R_{ia}}} \tag{1}$$

We merely assume that, when a new client joins the network, it can measure the loads of all APs within its hearing range. If the client does not affect an AP, or does so with negligible influence, it does not need to know the load on that AP. For example, if a client is far away from an AP, the interaction between them or the influence would be marginal and the client will not consider the information on that AP when it makes its AP association decision. We will show by implementation in section 5 that this assumption can be approximately achieved through a practical and low-overhead measurement method. We do not even assume how the loads will be changed when a client joins – although we do assume the load on each AP will be non-decreasing when a client joins. In the following, we examine several scenarios to show the ramifications of our assumptions and demonstrate how much our assumptions conform to the reality.

- *Interference with APs*: When client i joins the network, it might interfere with the transmission and reception on several APs. We denote the loads imposed on AP j after i makes its association decision as $\mathbb{L}_{ij}$. Note that j may not be the AP that client i associates with.
- *Interference with clients*: When client i joins the network, it might interfere with another client k. Even though i may not directly interfere with the AP (say AP j) that k is associated with (possibly due to being out of transmission range), the interference of i on k's communication may change the load on AP j. If the load on AP j is visible to client i, this scenario is amenable to our analysis; otherwise, we will ignore the load imposed by this indirect influence because the load change due to this rippling effect is marginal.
- *Myopic network configuration*: When a client i joins the network, it may not see all the APs because of the limited communication range. If the client does not see an AP, we assume the load change on that AP owing to the joining of client i is negligible. Furthermore, in this case, client i will not be able to associate with that AP because there is no usable bidirectional link between the client and the AP.

In summary, we study a practical and complicated wireless LAN model. The weight (communication load) a user adds on the associated AP might vary as the local network configuration changes or new incoming clients appear. Additionally, unknown number of new users may show up at any time. Therefore, an online algorithm is more suitable for this association situation.

Online algorithm description. Based on above model, we propose an online AP selection algorithm that is simple without a complex interaction among clients and APs. It runs as below. When a new client appears (in online fashion), it will make an irrevocable association with one of the visible APs so that the $\mathcal{L}_p$ norm of the loads on all the APs within its transmission range, after its join, will be minimized at the moment. Here $\mathcal{L}_p$ norm of loads $\mathcal{L}_1$, $\mathcal{L}_2$, ..., $\mathcal{L}_m$ is defined as $(\mathcal{L}_1^p + \mathcal{L}_2^p + \cdots + \mathcal{L}_m^p)^{1/p}$. By following this operation, the whole network can achieve good proven performance that is independent

of the number of incoming clients. Thus, re-association is not necessary for all clients already joined the network when a new client comes.

4.1 Performance Analysis

We first give a formal definition of the problem for the sake of quantitative analysis. In this realistic problem, we show that there is no constant approximation solution. Then we prove that competitive ratio of our online algorithm is $e \log m$, almost constant given a small number of APs in the network.

We first formalize this problem. Due to interference, a user may increase load to other APs, in addition to the AP that it associates with. To capture such interference effects, we denote by $\delta_{i,j,k}$ the increased load to AP j if user i chooses AP k. The generalized AP association problem (GAS) can be formulated as an integer linear programming problem.

$$\begin{aligned} \min_{\mathbf{x}} \max_{j} \quad & \sum_{i,k} x_{i,k}\delta_{i,j,k} \\ \text{subject to} \quad & \\ & \sum_{k} x_{i,k} = 1, \quad \forall i \in U \\ & x_{i,k} \in \{0,1\}, \quad \forall i \in U, k \in M \end{aligned} \tag{GAS}$$

where U is the set of users, M is the AP set, and $\mathbf{x} \in \{0,1\}^{|U|\times|M|}$ is the association matrix with elements $x_{i,k}$. The two constraints force each user to associate with exactly one AP.

In the above problem definition, for a specific user i and an AP j, even if i is not associated with j, the exact load induced to j is dependent on which AP user i associates with. Specifically, $\delta_{i,j,k}$ may be different from $\delta_{i,j,l}$ for $k \neq l$. We do not assume any relationship between different $\delta_{i,j,k}$, only assuming that they are non-negative (a new association won't decrease any AP's load). To see the difficulty of this problem, we should note that the input information can no longer be modeled by a bipartite graph.

In fact, this problem is a generalized version of the unrelated machine load balancing problem. Therefore, all the hardness results of this classic problem trivially hold for our problem. Specifically, no polynomial algorithm can achieve a competitive ratio less than 3/2 unless $P = NP$ [28]. No online algorithm can achieve a competitive ratio less than $\lceil \log(m+1) \rceil$ [1].*

The load balancing problem is able to be solved with constant approximation by linear relaxations [27], [28], but these two techniques are not effective in our case. [27] is not applicable to our problem. It requires to sort the users by their loads to the same AP, while in our case, one user may contribute different loads to the same AP. The algorithm in [28] is applicable to our problem, but it is $\Omega(m)$-competitive, not a constant approximation algorithm, as shown in the following theorem.

Theorem 1: To solve problem GAS, algorithm [28] is $(m+1)$-competitive where m is the number of APs. The bound is almost tight in that there exist a class of GAS instances that the algorithm gives m-competitive solutions.

The proof is presented in Section 2 of the supplemental file, which is available online. The main factor leading to this result is the multi-dimension nature of GAS.

Since the known constant approximation algorithms in load balancing literature fail to provide constant approximation guarantee for our problem, it is natural to ask whether constant approximation algorithm exists. Answering this question leads to the work of [31], and we report it in a separate work due to independent interest.

Theorem 2 ([31]): For any constant $c > 1$, there does not exist a polynomial time algorithm that is c-competitive for GAS, unless $P = NP$.

In this work, we are more interested in online algorithms, since users in a wireless network may come at any time. Currently, one of the best online algorithm for load balancing on unrelated machines is due to [32]. Their solution is $e \log m$-competitive. We show in the following that this algorithm is still applicable to our problem. More importantly, we prove that competitive ratio is still $e \log m$. This result is surprising in that (1) the competitive ratio is irrelevant of $\delta_{i,j,k}$ (2) the generalization does not increase difficulty for online algorithms. In addition, this competitive ratio is not far from the best we can do (up to a multiplicative constant e) for an online algorithm, since no online algorithm can do better than $\lceil \log(m+1) \rceil$ as mentioned before.

Since the client is unable to affect the other APs that are not in its hearing range, this algorithm will minimize the $\mathcal{L}_p$ norm of all the APs' loads in the system $(\mathcal{L}_1^p + \mathcal{L}_2^p + \cdots + \mathcal{L}_m^p)^{1/p}$ in each new association event. Then we have

Theorem 3: If the protocol is to minimize the $\mathcal{L}_p$ norm of the loads (rather than to minimize the maximum load), then the online protocol gives a $r \leq \frac{1}{2^{1/p}-1}$-competitive ratio.

The proof is presented in Section 2 of the supplemental file. The intuition is that a sub-optimal solution is achievable in the end by making the local-optimal association decision for each user that comes in the online manner.

It is worth mentioning that the competitive ratio in Theorem 3 is for minimizing the $\mathcal{L}_p$ norm of the loads. We need to transform this competitive ratio to be for minimizing the maximum load. Additionally, we need to find a p that gives reasonable competitive ratio. This is done by the following theorem.

Theorem 4: The online protocol is a $e \log m$ competitive protocol for minimizing the maximum load (or $\frac{1}{e \log m}$, w.r.t. maximizing the minimum throughput). This is obtained by setting $p = \ln m$ in the online protocol.

*. Their work considers a special case of unrelated machines. The machines have identical speed but each user can only select a subset of the machines. This case can be easily encoded by our model.

The proof is presented in Section 2 of the supplemental file.

Instead of being related to the number of APs and the ratio between the maximum and minimum rates, the competitive ratio of this protocol is linear to the logarithm of the number of APs, an almost constant competitive ratio for a small number of APs, which is deemed very promising since a constant competitive ratio algorithm usually gives a very good practical performance.

Furthermore, this algorithm has the advantage of computational simplicity and feasibility for practical implementation. The expected performance bound, for each client joined, is ensured just by the local network information at the moment it was coming as a new client. It is not necessary to reconsider its decision once a new association event occurs. In other words, our online algorithm takes exactly n steps to finish. In the next section, we demonstrate that this algorithm can be employed as a practical and light-weighted association protocol for off-the-shelf wireless LAN adapters.

5 Protocol Implementation of SmartAssoc

In this section, we describe how to efficiently realize SmartAssoc with low costs. Although aforementioned online algorithm is carefully designed to avoid large overhead caused by the measurement of nm^2 input parameters $\delta_{i,j,k}$ †, SmartAssoc still needs to address the real-world issue before we apply it in practice. That is how to estimate APs' load within a short time frame. It should not require any modification at the infrastructure side, so our implementation can be used in any open 802.11 networks or networks the client is able to access. To solve this problem, we propose a distributed light-weight AP load probing method in our SmartAssoc implementation, so that message exchanges between users and/or APs are not necessary. The load information is obtained by the user from target AP without association.

We consider the scenario that wireless link is the bottleneck of a communication connection. The discussion of other cases is out of the scope of this paper. Thus, the workload of an AP is reflected by the wireless traffic on air for this AP. Here we monitor the uplink stream traffic, ignoring the downstream, because accuracy improvement of throughput measurement is small compared with the extra complexity in the implementation experience of [25]. Every channel is considered to be interference-free with others, as this type of interference is ignorable compared to interference inside the channel. Thus, the computation of the $\mathcal{L}_p$ norm of the AP's workload can be reduced to per-channel based computation, while the comparison is still among all channels.

The implementation of SmartAssoc is taken on the popular commercial wireless LAN adapter by taking advantage of the legacy standard 802.11 protocol. User space control is provided by the Click Modular Router toolkit [33], while association functionality of the MadWifi driver v0.9.4 [34] is directly taken over by SmartAssoc in the monitor model.

As mentioned above, SmartAssoc requires to measure every AP's load on the same channel when a client i joins a candidate AP j ‡. A natural way to obtain this information is to let the client i perform an association operation with j, and generate traffic on j while at the same time capturing an uplink data stream for each AP by passive listening. The packet retransmission and duplication does not count. However, the association process consumes a lot of time, especially for encrypted wireless networks. When the set of association candidates is not small, the user is not able to bear waiting so long. Nevertheless, sending data packets without association will lead to rejection from the object AP because of the IEEE 802.11 standard. Thus, we find a more light-weight way to obtain the equivalent load information without association. Currently 802.11 standards require an AP to respond to *probe requests*, even if the request is sent by a station not associated with the AP §. We leverage this to create a packet type to replace the real data packet in the MAC layer. The intuition is to generate modified probe request traffic to the object AP j, similar to the data traffic, to estimate other APs' loads as if the client i is associating with j. The detail modifications of the probe request packet are made as follows.

- We make the probe request uni-cast, forcing the target AP to return an ACK upon receiving the packet. This behavior is similar to a station transmitting data packets to an AP. And this process is important as well for calculating the throughput.
- We change the subtype flag in the packet header to prevent the AP from returning a probe response, in order not to introduce unnecessary traffic to the network.
- The packet size, transmission rate, and inter-arrival time of modified probe requests are packet-wisely customizable by the user, which is able to provide more accurate throughput information for specific estimation based on upper-layer applications. This feature is implemented in the probing generator module. Its performance is shown in the next section.

We also implement an AP filter to make the candidate AP list programmable. The user can select a preferred

†. According to the online algorithm, an incoming user only needs to measure the resulting aggregate loads at APs. For example, during the i-th user's association, it checks all APs to find the best one. If it checks AP k, then he needs to measure $\delta_{i,j,k} + \sum_{i',k'} x_{i',k'}\delta_{i',j,k'}$ at all AP j where $i' = 1, \ldots, i-1$ are the previous users and $x_{i',k'}$ indicate their associations. This measurement can be done without measuring $\delta_{i,j,k}$ separately as illustrated in the rest of this section.

‡. Assume i always has some communication demand after association

§. It is done automatically by the firmware, transparent to the upper layers

channel, network, and minimal RSSI threshold to customize the list. Only qualified APs will be considered for estimation to reduce overhead.

The implemented protocol is described as pseudo code (Algorithm 1). A list of candidate APs is determined in the first place for estimation according to the beacons. In the list, the candidates from the same channel i group as a set $\mathcal{C}_i$. For each target AP j in $\mathcal{C}_i$, the user injects a probing traffic, which consists of modified probe request packets, to the AP j, while measuring all members' loads. After these measurements, the user can calculate the $\mathcal{L}_p$ norm loads for all members within this channel set, $(\sum_{k\in\mathcal{C}_i}\mathcal{L}_k^p)^{1/p}$, where p is the natural logarithm of the number of APs. This $\mathcal{L}_p$ norm value, influenced by association with the target AP, will be compared with the current best candidate AP among all channels. The comparison strategy applied in the best candidate updating stage is controlled through two programmable parameters. The first one is the norm-difference threshold T_{nd}, and the second one is RSSI-difference threshold T_{rd}. If the norm for the target AP are at least T_{nd} smaller than the norm for the current best AP, the target AP will be the best candidate AP instead of current one. Otherwise, the algorithm continues to check whether this norm is just smaller than the current best candidate's, as well as whether the target AP's RSSI is at least T_{rd} more than the best candidate's. If so, the target AP will be the best; for all other cases, we keep the current best AP without updating. The user treats every member of a channel set as the target member respectively, and repeats this process for each channel set. After the evaluation of all candidate APs, the user will associate with the final best candidate AP.

Algorithm 1 SmartAssoc Protocol Description

```
Discovers all available APs
Applies the filter on discovered APs
Puts all filtered APs into candidate list
for each C_i do
  for each AP j in C_i do
    Generates a probing traffic to j
    Estimates the new load of each AP within C_i
    Calculates the L_p norm change
    Updates the best candidate AP according to T_nd and
    T_rd
  end for
end for
Associates with the best candidate AP
```

6 Evaluation

We verify the feasibility of our online algorithm and demonstrate the performance of SmartAssoc implementation in this section. Each client is powered by a 1.66GHz CPU with 1 GB RAM, running on Linux kernel 2.26.24. A D-Link WNA-2330 with Atheros 5312 chipset wireless card is used.

6.1 Application Aware Probing

Since our modified probing stream, used to emulate the real data stream, is programmable in terms of the packet size, inter-arrival time, and transmission rate, it is easy to generate specific streams to mimic the data stream of a certain application. Thus, the client is able to find out the "best" association AP with respect to the application it wants to use. Figure 1 demonstrates how similar our probing can be to the secure copy (SCP) and VoIP protocols, respectively. The SCP protocol used is the Unix *scp* command line program. SCP transfers a single file from a laptop to a remote desktop on the Internet through a commercial AP on the channel 6 with RSSI -58. The packet size of the probe emulating SCP is 1500 bytes, and the inter-arrival time is presented at the right of Figure 1. On the left side of Figure 1, we choose Skype as the VoIP application to set up a communication between two laptops through the same commercial AP on the channel 1 with RSSI -33. The probing packet used in this experiment is 200 bytes on average. For both experiments, we first brought up our driver module to create virtual interfaces in the kernel for all upper-layer applications. Then a script was executed to associate with the target AP in each experiment. After association and IP assignment, we ran the application and our probing generator to generate the traffic, respectively. The traffic traces are captured by Wireshark for cumulative distribution function (CDF) calculation of the inter-arrival time. In these two experiments, the transmission rate for all streams is fixed at 36Mbps.

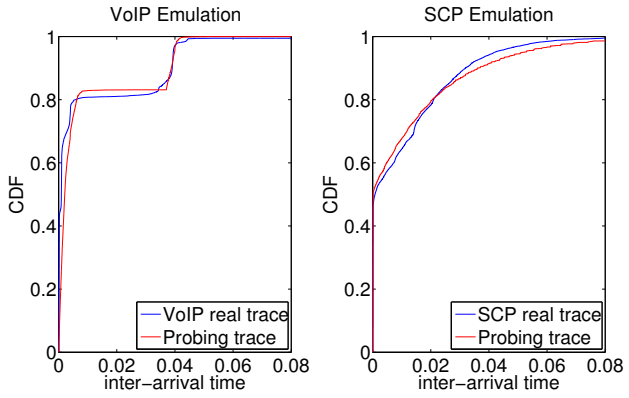

Fig. 1: Upper-layer application stream emulations.

6.2 Measurement Accuracy

The APs' load information needed in SmartAssoc is derived by monitoring the wireless channels. Although it is not necessary for monitoring to capture all packets on the air, a relatively accurate measurement can help to make a better association decision. Thus we conducted a series of experiments to investigate the capture missing, which is the main factor to cause the measurement error of the load. In this paper, we are focusing on the Atheros 5212 chipset, while other chipsets can be easily studied like this as well. We set up two laptops with a distance of

10 feet between them. One is the target laptop, which is used to generate a data stream for measurement. The other laptop acts as the monitor to estimate the data throughput from the target laptop. To make our experiment comprehensive, we use different transmission rates when transmitting the data streams. The rates used are 1Mbps, 2Mbps, 5.5Mbps, 11Mbps, 18Mbps, 36Mbps, and 54Mbps. We also use different inter-packet times of 5ms, 10ms, 15ms, and 20ms at each rate, respectively. The packet size in all trials is 1000 bytes, and every trial last 5 seconds for data stream generating and capturing. The experiment results are shown in Figure 2. The x axis presents the captured packet number in each trial at the monitor laptop side, while the y axis presents the number of transmitted packets counted at the target laptop side. It is clear that if there is no error, all points (star or circle) should fall on the green dash line $y = x$. Based on these experimental results (excluding the six outliers), we are able to calculate the best linear fitting by using the Least Square method, which is shown as the red line, $y = 0.8806 \times x$. 0.8806 is used for estimation calibration with respect to the Atheros 5212 chipset, i.e., $estimation = \frac{measurement}{0.8806}$

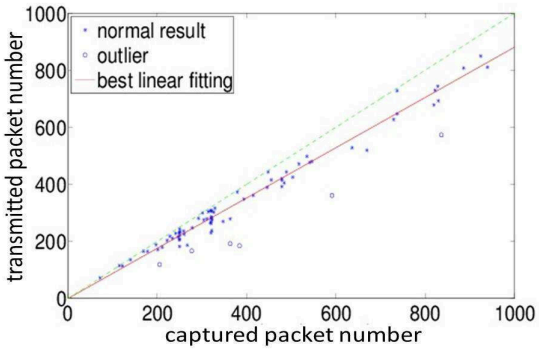

Fig. 2: Calibration Experiments for Atheros 5212 chipset

6.3 Measurement Duration

The long measurement time of channel traffic, although it benefits the accuracy of workload estimations, extends the delay time before associations and consumes more power. Thus, we conducted experiments to find out what is the minimum measurement duration given certain measurement error. To facilitate experiments, we implemented a testing program based on `libpcap`. It has two functionalities. The first one is a standalone component of AP workload estimation described in our protocol. The second one is a wireless traffic generator that can assemblies 802.11 packets of our choice and transmits them at given traffic patterns. Two laptops installed this testing program are set in 5.18 GHz of 802.11a. One of them is used to generate a traffic to one AP set in the same channel. The inter-arrival time of generated wireless traffic follows the exponential and normal distributions respectively. The other one is to estimate this AP's workload in different time windows. The experimental results are illustrated in Fig 3 and Fig 4. We found out 50 ms is a proper duration that can achieve less than 3 percent estimation errors.

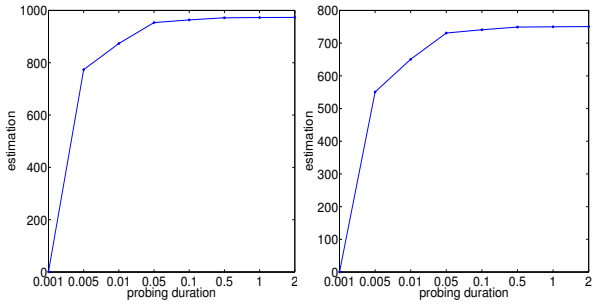

Fig. 3: Estimation of workloads generated by two traffic patterns following exponential distributions with mean of $\frac{1}{973}$ and $\frac{1}{751}$ respectively. X axis is different traffic measurement time from 0.001 seconds to 2 seconds. Y axis is the estimated workloads in packets per second.

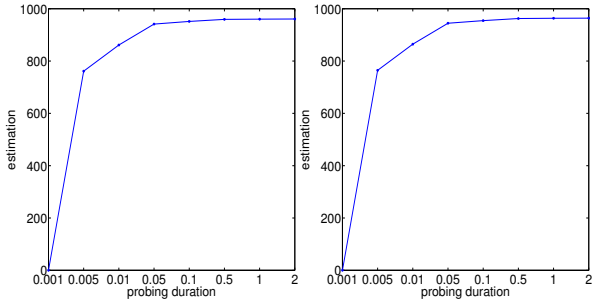

Fig. 4: Estimation of workloads generated by two traffic patterns following normal distributions. In the left figure, the mean of distribution is 961 with standard deviation of 1, while in the right one, the mean is 964 with standard deviation of 4. X axis is different traffic measurement time from 0.001 seconds to 2 seconds. Y axis is the estimated workloads in packets per second.

6.4 Comparison Experiment

We conduct an experiment to compare our association method of SmartAssoc with other practical ones, the Best-throughput and Best-RSSI strategies. Best-throughput here is one special case of the selfish strategy, of which the convergence speed can be bad. It means every client will make an irrevocable association decision to maximize its own throughput. In the experiment, there are three APs consisting of an *extended service set* (ESS) and four wireless LAN clients. Two of the APs (AP-1 and AP-3), whose process capabilities are relatively stronger than the thirds , are set in channels 1 and 11, respectively, and the third one (AP-2) is set in channel 11 as well. Three of the four clients, STA-1, STA-2 and STA-4, are put close to each other. Detailed settings are shown in Table 1.

The experiment includes four trials. In each trial, clients came to join the ESS one by one by using the same association strategy. It is reasonable because that, in real world, the time to perform the association operation is statistically much smaller than the inter-arrival time

TABLE 1: Comparison Experiment Settings

APs				
ID	Model		Channel	
AP-1	LinkSys WRT54G		CH 11	
AP-2	D-Link DI-713P		CH 11	
AP3	LinkSys WRT54G		CH 1	
Clients				
ID	Adapter	Pkt payload (byte)	Inter-pkt time(ms)	Transmission rate(Mbps)
STA-1	internal	1000	5	2
STA-2	internal	1000	5	11
STA-3	external	1000	5	11
STA-4	external	1000	5	11

between new users, so the possibility that two clients will want to join the ESS at the same time is low. After joining, the client will generate traffic using the configuration in table 1 to the associated AP. A trial was repeated three times, one for each association strategy.

The minimum client throughput for each strategy in the four trials are presented in Figure 5. Our algorithm performs better than the other two because it can balance the APs' workloads and reduce the interference among all wireless nodes. The performance of the Best-throughput is unstable, and it also indicates that the selfish strategy cannot compete with ours under similar overhead costs. Meanwhile the Best-RSSI strategy often makes all clients associate with the same AP.

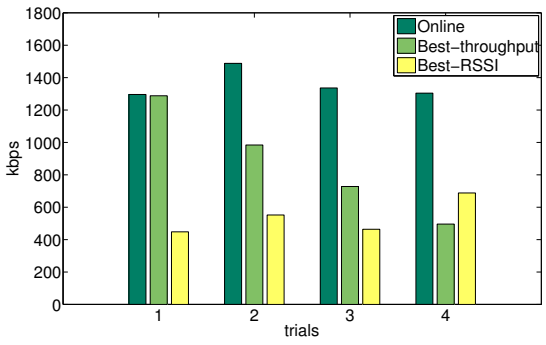

Fig. 5: Comparison Experiment Results

6.5 Overhead

All three protocols mentioned above need an AP discovery phase. However, Best-RSSI does not have any other overhead, whereas Best-throughput and our protocol need extra time to evaluate every discovered AP. For each channel, our protocol only spends a small amount of time on the channel measurement, compared with the Best-throughput one. Nevertheless, the Best-throughput protocol requires real de-association and re-association operations, including the IP assignment, before every measurement. These operations consume a lot of time, from 3 sec to 8 sec. When the AP number grows, the Best-throughput protocol spends much more time than ours. Therefore, the overhead of the selfish protocol, which is equivalent to multiple runs of the Best-throughput protocol, is even worse. Thus, our protocol is the most efficient one.

We further examine the systematic impact of our implementation on mobile devices in terms of energy consumption and time overhead. SmartAssoc protocol is performed on the smartphone that is one of most energy-constrained mobile devices, and at the same time energy cost is monitored by external power meter (Fig 6). Experimental results show that, given the probing packet size is 1000 bytes, the average power consumption is 892 mW and the standard deviation is 57 mW. Provided that there are five APs within the smartphone's transmission range, the total energy cost is 164 uAh, which is only 0.01% of one typical 1500-mAh smartphone battery. The time overhead is 2.5 seconds. The user might not notice this delay if SmartAssoc is triggered with the AP scanning procedure of 802.11 right after the wake-up of phone.

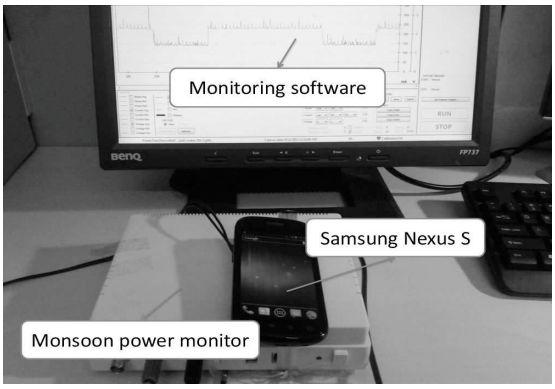

Fig. 6: Energy consumption experimental setting.

6.6 Scalability

Scalability is the capability of a protocol to handle in the scenario of growing network, i.e., more users and APs. Unlike the centralized approach, the online algorithm applied in our protocol runs independently on each client and does not need any assistance or collaboration from each others, so more users will not introduce more extra overhead compared with less-user case. Besides, no workload pressure is added to the infrastructure since there is no central server behind the network. According to our algorithm design, the client makes its association decision only based upon load information of APs within its transmission range. Therefore, more APs does not mean more complicated association procedure from the perspective of a client. Additionally, newly joined clients will not affect any already associated client, avoiding decision revocations that introduce complexity and overhead.

Our implementation also makes the association protocol scalable. There is no extra cost on the infrastructure compared with standard 802.11 network, and users still can use their commercial chipsets. We also present the simulation results of our protocol in larger network settings in the next section.

7 Simulation Result

We use simulations to evaluate our association protocol SmartAssoc on a larger scale with more wireless nodes and various configurations. We use NS2 version 2.33 as our simulator. The multiple channel feature is patched into the NS2 wireless portion following the instructions of [35]. The MAC layer type is 802.11, while the radio propagation model is two-ray-ground. Ad-Hoc routing protocol is disabled since we are focusing on the infrastructure type of wireless LAN. The RTS/CTS mechanism is also disabled. The data traffic for users is a UDP stream with a packet size of 1000 bytes and average inter-arrival time of 1 ms. The transmission rate is set to 11 Mbps. The selected channels include 1, 4, 5, 6 , and 11 for covering the orthogonal and adjacent channel cases. The throughput measurements are between the wireless nodes.

We implement the following three practical association protocols for comparison.

1) **Online**. This is our proposed online algorithm in SmartAssoc. It is implemented as described in Algorithm 1.
2) **Selfish**. The behavior of *Selfish* is that every client always has the incentive to change its association decision if it can find an AP providing higher throughput than current one. Since the convergence speed of selfish strategy can be very bad, we demonstratively run the protocol for 5 rounds. In the first round, the clients come to join the wireless LAN one by one. Each client will associate with every AP to measure the UDP throughput and pick the one who is able to offer the highest value. In each of the next four rounds, every client will repeat the above process to adjust its association based on current wireless LAN association topology. Finally, every client will keep its association with the AP it picks in the last round.
3) **Ideal**. This is the globally optimal association solution in terms of maximizing the minimum throughput over all clients. *Ideal* is obtained by enumerating all possible association topologies, given a specific scenario setting only including the location and channel assignment information. In the real world, it is not practical because of its complexity.

Every experiment conducted below consists of many trials. Each trial has its own scenario configuration. The configuration provides the locations of all wireless nodes and the APs' channels. Both of these two information are randomly generated. Every throughput measurement, no matter whether it is for a data stream or probing stream, takes 3 seconds in the simulation.

The first simulation is to study the competitive ratio of *online* compared with *ideal* from the perspective of empirical experiments. The experimental value of the competitive ratio is a good indicator of the performance gap on average between the *online* and *ideal*. The statistically stable performance is also a concerned issue to the users in the real world. This experiment randomly picked 50 scenarios for testing. For every scenario, the competitive ratio in terms of the minimum client throughput, shown in Figure 7, is calculated based on the test result of *online* and *ideal*. The theoretical upper bound is also provided for comparison. The simulation results shows that about 86 percent of competitive ratio is above 0.47, and 70% are quite stable, just around 0.5. The worst competitive ratio among these 50 trials is 0.313, while the theoretical upper bound, computed from $\frac{1}{e \log m}$, is 0.232.

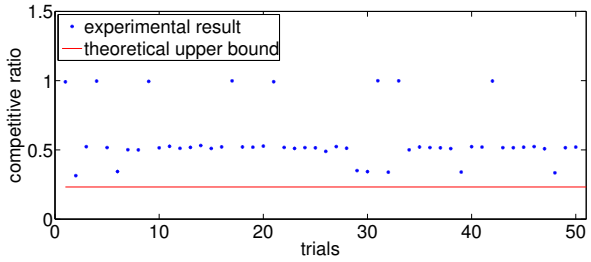

Fig. 7: Competitive ratio results, w.r.t. minimum client throughput, for 5 clients and 3 APs within 20×20. The simulation repeated 50 times with different configurations.

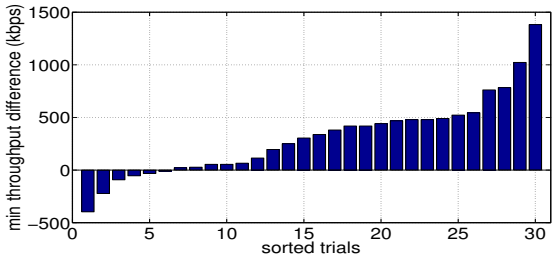

Fig. 8: Simulation result for 10 clients and 3 APs within 30×30. The simulation repeated 30 times. Each bar is the representative of minimum throughput difference for each trial. The results are sorted in ascending order

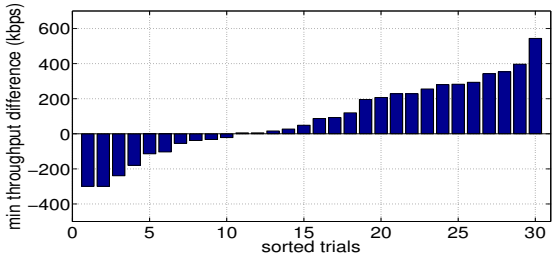

Fig. 9: Simulation result for 20 clients and 6 APs within 90×90. The simulation repeated 30 times. Each bar is the representative of minimum throughput difference for each trial. The results are sorted in ascending order

Next, we conducted a scale-up comparison simulation between *online* and *selfish*, which includes three experiments to show the performance in large scale deployments. In the first experiment, there are 10 clients and 3 APs within a rectangle of 30×30; the second one has 20 clients and 6 APs located in a rectangle of 90×90; 30 clients and 9 APs are involved in the third experiment within a rectangle of 150×150. Each experiment ran 30 trials. For each trial scenario, both our strategy and the selfish strategy were applied for the association

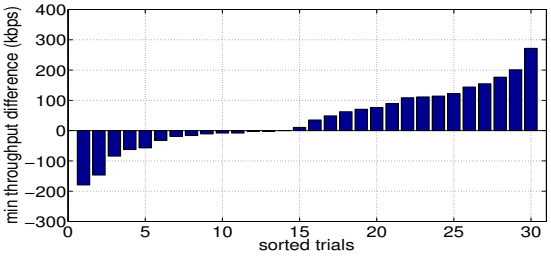

Fig. 10: Simulation result for 30 clients and 9 APs within 150×150. The simulation repeated 30 times. Each bar is the representative of minimum throughput difference for each trial. The results are sorted in ascending order

processes of all clients on this setting, respectively. After finishing all users' association processes, we measured the UDP traffic throughput for every client and found the minimum, T_{online} for *online* and $T_{selfish}$ for the *selfish* strategy. Then the minimum client throughput difference, shown in Figures 8, 9, and 10, is calculated by using $diff = T_{online} - T_{selfish}$. These figures show that, even through the selfish protocol is allowed to consume more time, our strategy is more often to perform better in terms of maximizing the minimum client throughput.

In the *online* strategy, since every client only needs to run our association once, the following clients in the future will not affect the behaviors of current associated clients. Meanwhile, for the *selfish* protocol, the unexpected new clients can easily break the current equilibrium into an unstable state, which will interrupt the usage of users. Thus, the *online* is more practical and less-intrusive. From the figures, it is shown that ours can, despite not knowing who will come to join the network, reduce the performance downgrade for the client who has the minimum throughput.

8 Conclusion

In this paper, we consider the problem of AP association in WLAN. We present theoretical analysis of the performance of two commonly used AP selection protocols, and propose an online algorithm with a provable good competitive ratio. The association protocol based on this algorithm is implemented on the real testbed in a lightweight way. We evaluated our scheme using a combination of implementation on commodity hardware and extensive simulation. We demonstrate that it is promising that by combining our theoretical understanding and real system implementation experience, a new, practical, and better AP association protocol is possible.

References

[1] Y. Azar, J. Naor, and R. Rom, "The competitiveness of on-line assignments," *J. Algorithms*, vol. 18, no. 2, 1995.

[2] V. Mhatre and K. Papagiannaki, "Using smart triggers for improved user performance in 802.11 wireless networks," in *MobiSys 2006*.

[3] A. J. Nicholson, Y. Chawathe, M. Y. Chen, B. D. Noble, and D. Wetherall, "Improved access point selection," in *MobiSys 2006*.

[4] K. Sundaresan and K. Papagiannaki, "The need for cross-layer information in access point selection algorithms," in *IMC 2006*.

[5] S. Shakkottai, E. Altman, and A. Kumar, "Multihoming of users to access points in wlans: A population game perspective," *IEEE JSAC 2007*.

[6] T. Korakis, O. Ercetin, S. Krishnamurthy, L. Tassiulas, and S. Tripathi, "Link Quality based Association Mechanism in IEEE 802.11h compliant Wireless LANs," in *RAWNET 2005*.

[7] H. Wu, K. Tan, Y. Zhang, and Q. Zhang, "Proactive scan: Fast handoff with smart triggers for 802.11 wireless lan," *INFOCOM 2007*.

[8] A. Mishra, M. Shin, and W. Arbaugh, "An empirical analysis of the ieee 802.11 mac layer handoff process," *SIGCOMM Computer Communication Review 2003*.

[9] I. Ramani and S. Savage, "Syncscan: practical fast handoff for 802.11 infrastructure networks," in *INFOCOM 2005*.

[10] M. Lu and J. Wu, "Localized Access Point Association in Wireless LANs with Bounded Approximation Ratio," in *ICCCN 2008*.

[11] H. Han, B. Sheng, C. C. Tan, Q. Li, and S. Lu, "A measurement based rogue ap detection scheme," in *INFOCOM*, 2009.

[12] F. Xu, C. Tan, Q. Li, G. Yan, and J. Wu, "Designing a practical access point association protocol," in *INFOCOM*, 2010.

[13] G. Judd and P. Steenkiste, "Fixing 802.11 access point selection," *ACM SIGCOMM Computer Communication Review*, 2002.

[14] S. Vasudevan, K. Papagiannaki, C. Diot, J. Kurose, and D. Towsley, "Facilitating access point selection in ieee 802.11 wireless networks," in *IMC 2005*.

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

[16] Cisco System Inc., "Data sheet for cisco aironet 1200 series," 2004.

[17] S. Suri, C. Tóth, and Y. Zhou, "Selfish load balancing and atomic congestion games," *Algorithmica*, vol. 47, no. 1, pp. 79–96, 2007.

[18] I. Milchtaich, "Congestion games with player-specific payoff functions," *Games and economic behavior*, vol. 13, no. 1, pp. 111–124, 1996.

[19] E. Even-Dar, A. Kesselman, and Y. Mansour, "Convergence time to Nash equilibria," *Lecture Notes in Computer Science*, pp. 502–513, 2003.

[20] K. Mittal, E. Belding, and S. Suri, "A game-theoretic analysis of wireless access point selection by mobile users," *Computer Communications*, 2008.

[21] M. Cesana, I. Malanchini, and A. Capone, "Modelling network selection and resource allocation in wireless access networks with non-cooperative games," in *IEEE MASS*, 2008.

[22] Y. Bejerano, S.-J. Han, and L. Li, "Fairness and load balancing in wireless LANs using association control," *ACM/IEEE Transactions on Networking*, 2007.

[23] R. Murty, J. Padhye, A. Wolman, and B. Zill, "Designing high performance enterprise wi-fi networks," in *NSDI 2008*.

[24] R. Chandra, P. Bahl, and P. Bahl, "Multinet: connecting to multiple ieee 802.11 networks using a single wireless card," in *INFOCOM 2004*.

[25] S. Kandula, K. C.-J. Lin, T. Badirkhanli, and D. Katabi, "Fatvap: Aggregating ap backhaul capacity to maximize throughput," in *NSDI 2008*.

[26] Y. Wang, Y. He, X. Mao, Y. Liu, Z. Huang, and X. Li, "Exploiting constructive interference for scalable flooding in wireless sensor network," in *INFOCOM'12*.

[27] D. B. Shmoys and E. Tardos, "An approximation algorithm for the generalized assignment problem," *Math. Program.*, vol. 62, December 1993.

[28] J. K. Lenstra, D. B. Shmoys, and E. Tardos, "Approximation algorithms for scheduling unrelated parallel machines," *Math. Program.*, vol. 46, February 1990.

[29] D. Luo, X. Zhu, X. Wu, and G. Chen, "Maximizing lifetime for the shortest path aggregation tree in wireless sensor networks," in *INFOCOM'11*, pp. 1566 –1574.

[30] G. Matthew, *802.11 wireless networks: the definitive guide*, 2002.

[31] X. Zhu, Q. Li, W. Mao, and G. Chen, "Online vector scheduling and generalized load balancing," *Technical Report WM-CS-2012-01*, 2012.

[32] I. Caragiannis, "Better bounds for online load balancing on unrelated machines," in *SODA 2008*.

[33] "http://read.cs.ucla.edu/click/."

[34] "http://madwifi-project.org/."

[35] A. Raniwala and T. Chiueh, "Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network," in *INFOCOM 2005*.