CACAO: Distributed Client-Assisted Channel Assignment Optimization for Uncoordinated WLANs

Xiaonan Yue, Chi-Fai Michael Wong, and Shueng-Han Gary Chan

Abstract—IEEE 802.11 WLANs are becoming more and more popular in homes and urban areas. As compared to traditional WLAN setups (such as in campuses) where knowledgeable network administrators can make centralized decisions on channel selection, access points (APs) in these networks are often deployed by network nonspecialists in an uncoordinated manner, leading to unplanned topologies, interference, and therefore unsatisfactory throughput performance. We consider in this paper a distributed channel assignment algorithm for uncoordinated WLANs, where APs can self-configure their operating channels to minimize interference with adjacent APs. We first formulate the optimization problem on channel assignment which overcomes some of the weaknesses encountered by uncoordinated WLANs. We show that the problem is NP-hard, and propose an efficient, simple, and distributed algorithm termed CACAO (Client-Assisted Channel Assignment Optimization). In CACAO, the clients feed back traffic information to their APs. This leads to better network condition knowledge and better channel assignment decisions at the APs. We conduct extensive simulation study and comparisons using Network Simulator 2 (NS2). Our results show that CACAO outperforms other traditional and recent schemes in terms of TCP and UDP throughputs with a similar level of fairness. Furthermore, it converges quite fast and reduces cochannel interference significantly.

Index Terms—Wireless, channel assignment, client-assisted, traffic aware.

1 Introduction

In recent years, we have witnessed increasing penetration of wireless broadband Internet into our everyday lives. This is mainly due to the availability of affordable Wi-Fi services and related consumer products, such as laptops, PDAs, gaming devices, digital cameras, etc [1], [2]. In order to allow these devices to access the Internet, more and more people are setting up Wireless Local Area Networks (WLANs) in homes, offices, schools, shopping centers, libraries, airports, or wherever wireless connection is desirable [3]. A recent study indicates that the total number of new Wi-Fi enabled consumer electronic devices shipped will grow from 40 million in 2006 to nearly 249 million in 2011 [4]. Driven by the growth of these Wi-Fi enabled devices, demands for new WLANs are expected to rise quickly in the near future.

These WLANs share some common features. They tend to be *uncoordinated*, small in size, independently owned and managed, and deployed in areas where access points (AP) density may vary greatly. Although IEEE 802.11b/g defines a total of 14 channels, only three (namely, channels 1, 6, and 11) are nonoverlapping and can be used simultaneously without causing interference. With the limited number of nonoverlapping channels, channel assignment becomes a critical performance factor for uncoordinated WLANs. Compared to traditional WLAN deployments (e.g., enterprise or campus environment), uncoordinated WLAN deployments (e.g., residential environment) present the following challenges in the design of channel assignment algorithm:

- *Network nonspecialists.* Unlike WLANs managed by certified system administrators, uncoordinated WLANs are usually set up by inexperienced and independent users who are not knowledgeable in network configuration. These users very likely expect the devices to be plug-and-play (i.e., self-configurable). They cannot be expected to know how to configure the appropriate channel to minimize interference in their neighborhoods.
- *Unplanned topology.* In managed WLANs, such as campus or enterprise networks, the system administrators can calculate where the APs should be placed in order to minimize cochannel interference and achieve high throughput. In uncoordinated WLANs, on the other hand, APs are placed without any concerted planning; some areas may have high AP density and hence may experience high interference and poor performance.
- *AP independence.* Due to the absence of central management and the prevalence of inexperienced users, configuration for uncoordinated WLANs should be as simple and automatic as possible. The APs of different WLANs should operate independently without any direct communication.

As the number of uncoordinated WLANs increases, so does the cochannel interference. This is evident from the measurement experiments conducted by Akella et al. [5],

- *The authors are with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong.*
  E-mail: {csjohn, mwong, gchan}@cse.ust.hk.

Manuscript received 7 Oct. 2009; revised 6 Mar. 2010; accepted 13 Sept. 2010; published online 3 Feb. 2011.
Recommended for acceptance by L. Li.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2009-10-0493.
Digital Object Identifier no. 10.1109/TPDS.2011.59.

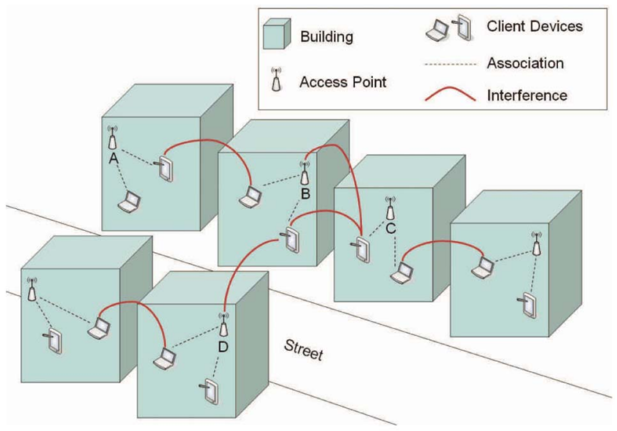


Fig. 1. Interference in uncoordinated WLANs.

which show that most of the deployed APs transmit on the same channel (channel 6), and only a small fraction (14 percent) of APs use the remaining two nonoverlapping channels. This means that many APs that overlap in coverage are not properly configured to reduce interference, which leads to low system throughput, as shown by Jain et al. [6]. Hua and Zheng [7] also show that there may even be starvation (i.e., very low throughput) in a dense 802.11 wireless network.

We show in Fig. 1 a typical scenario of uncoordinated WLANs in a residential environment, where each home sets up its own AP independently without any a priori agreement on channel assignment and AP placement. Devices of different networks interfering (i.e., within the transmission range) with each other are connected with a solid line. For channel autoconfiguration, APs traditionally use Least Congested Channel Search (LCCS), in which each AP scans all available channels and chooses the one used by the least number of associated devices (i.e., the so-called least "congested" one) as its operating channel. However, LCCS suffers from the hidden interference problem and the nonuniform traffic distribution problem (discussed in detail in Section 2). Consider the APs A, B, C, and D in Fig. 1. Even though they are not interfering with each other directly, their clients *are* in various ways. If these APs choose to use the same channel, there may be much interference, affecting the client traffic and the overall throughput.

In this paper, we propose and study a novel distributed channel assignment scheme for uncoordinated WLANs. The APs autoconfigure their channels depending on their local traffic information. Our approach, termed Client-Assisted Channel Assignment Optimization (CACAO), makes use of client feedback to perform channel assignment. Such feedback may be obtained using the proposed IEEE 802.11k standard for radio resource management, which defines a series of measurement requests and statistical reports between an AP and its clients [8].

CACAO is scalable, as it is completely distributed. In CACAO, mobile clients help their APs to detect interference from adjacent networks. APs periodically query their associated clients to collect reports on traffic statistics of each channel. With this information, they are able to get a better view of channel conditions, and hence can dynamically and automatically reconfigure themselves to operate on the "best" channel to reduce interference among basic service sets (BSSs). Our approach overcomes the traffic distribution problem and addresses the hidden interference problem.

We implement CACAO using NS-2 (version 2.30) with support of multiple nonoverlapping channels, and conduct extensive simulations. We compare it with the traditional LCCS, state-of-the-art MAXChop [9], and Hminmax [10]. Our simulations show that networks using CACAO experience much lower interference and achieve higher throughput compared to networks using traditional and recent schemes. CACAO also achieves superior throughput with similar fairness.

The rest of this paper is organized as follows: In Section 2, we survey related work. The problem formulation and CACAO algorithm are presented in Section 3. In Section 4, we discuss illustrative simulation results and comparisons. We conclude in Section 5. Detailed literature survey, some of the weaknesses of traditional channel assignment approaches, and the implementation issue of CACAO are presented in the supplementary file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2011.59.

2 Related Work

Much work has been done with the primary objective to reduce interference and increase system throughput. There are many centralized algorithms that assume network administrators conduct site surveys and do propagation modeling before network deployment [11], [12], [13], [14], [15]. However, in uncoordinated WLANs, we cannot rely on administrators to configure the network.

Mishra et al. propose a dynamic channel assignment algorithm called CFAssign-RaC to achieve load-balancing based on a "conflict set coloring" formulation [16]. Ahmed et al. propose an algorithm using successive refinement to solve a joint channel assignment and power control problem [17]. Kauffmann et al. propose a measurement-based self-organization approach for channel assignment [18]. All these approaches focus on networks where all the participating devices belong to the same enterprise, and hence cannot be applied to uncoordinated WLANs. Later proposed traffic-aware approaches in [19] and [20] consider the changing traffic patterns to make channel assignment decisions. Being traffic-aware, these approaches are able to reduce interference dramatically. However, they are still centralized algorithms and CACAO is able to be traffic-aware in a distributed manner.

Mishra et al. propose a distributed algorithm called MAXChop [9], which addresses channel assignment problem based on standard graph coloring formulation and calculates a channel hopping sequence at each AP to reduce interference. However, such hopping sequence needs to be periodically communicated among APs, and frequent hopping introduces much overhead into the system. Moreover, the MAXChop algorithm has not considered the changing traffic pattern. As a result, it is possible that heavily loaded adjacent APs are assigned to the same channel at some hopping slots, leading to high interference. Arbaugh et al. propose Hminmax [10] and formulate the channel assignment problem as a weighted coloring graph

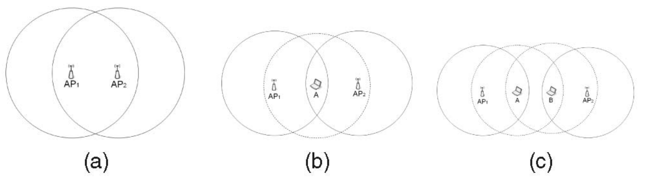


Fig. 2. The three interference cases in which two APs have an edge between themselves in the formulated graph problem. (a) Case 1. (b) Case 2. (c) Case 3.

problem. Their approach is based on the interference experienced by clients. CACAO differs by taking into account the real traffic load of *both* APs and clients, which leads to better interference mitigation and better performance. In this paper, we compare these two recent schemes (MAXChop and Hminmax) with CACAO.

3 CACAO Channel Assignment

We first present the problem formulation in Section 3.1 and show that it is NP-hard. We then propose CACAO, a distributed algorithm to achieve high throughput and fairness in Section 3.2.

3.1 Problem Formulation

In this paper, we consider uncoordinated WLANs with the following characteristics:

- The APs are placed in an uncoordinated manner; there is no central administrator to decide on the number of APs to be placed and where they should be deployed.
- Each AP belongs to and is managed by a different individual or organization. Since the APs belong to different authorities (which may have security and privacy concerns), load balancing by device association among APs is not allowed.
- Channel assignment decisions must be made based on only the information gathered by the AP and its associated clients.
- The clients associated with an AP only send data to or receive data from their own AP.

We model the network as a graph $G = (V, E)$, where $V = \{ap_1, ap_2, \ldots, ap_n\}$ is the set of n APs. In the interference map, there is an edge between ap_i and $ap_j (ap_i \neq ap_j)$ as long as two nodes from their corresponding BSSs interfere with each other. We illustrate three cases where two BSSs having an edge between each other in Fig. 2. The circle of a particular node indicates its interference range. Fig. 2a shows the case where two APs are within the interference range of each other, and hence directly interfere with each other. Fig. 2b shows two APs (AP_1 and AP_2) and client A, which is associated with AP_1. Although these two APs do not interfere with each other directly, AP_2 interferes with client A. Therefore, an edge exists between AP_1 and AP_2. In Fig. 2c, the two APs cannot detect the existence of each other's network directly. However, due to reports from clients A and B, AP_1 and AP_2 are aware of each other's existence. Therefore, there is an edge between these two APs.

In this model, clients associated with an AP are grouped with the AP and collapse into a single (super) node for the

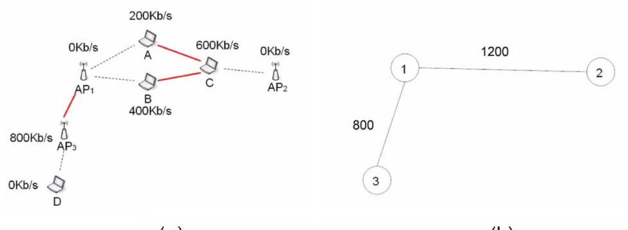


Fig. 3. An example to illustrate CACAO's weight calculation. Solid line indicates interfering traffic and dashed line indicates existing traffic. (a) An example topology with traffic information. (b) The resulted graph.

purpose of interference accounting. Let $W(ap_i, ap_j)$ be the potential interference level between AP_i and AP_j. The larger the weight $W(ap_i, ap_j)$ is, the higher the interference between AP_i and AP_j will be if they use the same channel.

A very important part of CACAO algorithm is to determine $W(ap_i, ap_j)$, which represents the level of possible interference between BSSs. This is also the part where client side channel condition report is used by APs to perform better channel assignment. The method we use is to sum up the total bitrate of each interfering flow belonging to two different BSSs. A and B observe 600 kb/s from C which is associated with a different AP. C, on the other hand, observes 200 kb/s and 400 kb/s from A and B, respectively. Therefore, the level of interfering traffic observed by ap_1 with regard to ap_2 is 1,200 (e.g., $600 + 200 + 400$), which is the edge weight $W(ap_1, ap_2)$. Similarly, $W(ap_1, ap_3)$ is 800. The resultant graph (with regard to ap_1) is shown in Fig. 3b.

Given the graph, let APS_i be the set of nodes associated with ap_i. Also let T_n be the bitrate of outgoing traffic from node n over a certain time interval. Then the potential interference level between AP_i and AP_j for this time interval can be calculated by

$$W(ap_i, ap_j) = \sum_{n \in APS_i, m \in APS_j, i \neq j,} (T_n + T_m), \tag{1}$$

where n and m interfere with each other.

Next we define a Boolean function interference map $I(ap_i, ap_j)$ for each edge, where

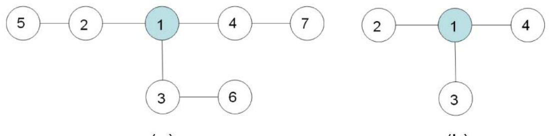
$$I(ap_i, ap_j) = \begin{cases} 1, & \text{if } ap_i \text{ and } ap_j \text{ are on the same channel;} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

One may consider that the product of $W(ap_i, ap_j)$ and $I(ap_i, ap_j)$ indicates the total interference level between ap_i's BSS and ap_j's BSS.

Given $G(V, E)$ and $W(ap_i, ap_j)$, channel assignment C is a mapping $C : V \to \{1 \ldots k\}$, where k is the total number of nonoverlapping channels (e.g., $k = 3$ for the IEEE 802.11b/g standard). The channel assignment problem is to find a mapping C such that the total interference level is minimized, i.e.,

$$\min_C L(G, C) = \sum_{\forall e = (ap_i, ap_j) \in E} W(ap_i, ap_j) \times I(ap_i, ap_j). \tag{3}$$

Theorem. *Given a weighted undirected graph $G = (V, E)$, with n vertices, $V = \{v_1, v_2, \ldots, v_n\}$, and a weight function*



(a) (b)

Fig. 4. An example to illustrate the definition of $G(ap_i)$. (a) An example of graph G. (b) The graph of $G(ap_1)$.

$W(v_i, v_j)$. Let $C : V \rightarrow \{1 \dots k\}$ be a k-coloring of G. The problem of finding C that minimizes

$$L_G = \sum_{\forall e=(v_i,v_j)\in E, C(v_i)=C(v_j)} W(v_i, v_j) \tag{4}$$

is NP-hard.

Proof. First, we consider the general problem of coloring an undirected graph, i.e., given a graph $G = (V, E)$, does there exist a k-coloring $C : v \rightarrow 1 \dots k$, for $1 \leq k \leq |V|$, such that no two adjacent vertexes are of the same color? By assigning a weight of 1 to all edges in our problem formulation, we see that the objective value $L = 0$ iff G has a k-coloring assignment. Thus, the hardness of our problem follows the hardness of the general graph coloring problem, which is NP-hard. □

Since the problem is NP-hard, we present a distributed algorithm CACAO that runs independently on each AP and client. The APs perform local optimization based on the objective function and seek to minimize the expected interference level for each time interval.

3.2 Distributed Algorithm

CACAO utilizes information gathered by APs and clients on interference conditions to compute and minimize a local objective function by switching to a channel that has least expected interference. As mentioned, the algorithm is very simple, and requires no direct communication between neighboring APs. For networks with stable traffic pattern, CACAO adaptively settles the global objective function value to a local minimum.

Let $ap_i.c$ denotes the operating channel of ap_i. Let $G(ap_i) = (V(ap_i), E(ap_i))$ be the subgraph of G containing only vertex ap_i and all its directly connected neighbors. We show in Fig. 4 an example to illustrate the definition of $G(ap_i)$. Fig. 4a is the complete graph which contains seven vertices. Vertices 2, 3, and 4 are directly connected to vertex 1. $G(ap_i)$ shown in Fig. 4b is the graph that ap_i can obtain locally based on channel condition reports from its clients. Since the measurement of each AP and its clients can only discover nearby nodes that are within the interference range, the weight calculation and channel assignment are done locally based on the graph $G(ap_i)$ for each ap_i.

We show the CACAO algorithm for ap_i in Algorithm 1. First, every AP runs the initialization routine when it boots up. During the boot-up period, each AP randomly assigns itself to a channel chosen from the k nonoverlapping channels (because there is no previous traffic information to use to assign channel). Next, each AP periodically runs the optimization routine to retrieve its clients' interference statistics and then computes the sum of the expected interference levels with regard to each nearby BSS for each channel for the next time interval. Because many online applications (e.g., video streaming) have steady data transfer rate, the amount of traffic observed for the current time interval is a proper estimation for the traffic in the next time interval. Therefore, after the optimization routine calculates and compares the expected interference level of each channel, the AP chooses the channel that yields the least expected interference, and switches to that channel. At the end of every time interval, each AP independently chooses the appropriate channel assignment to locally minimize the objective function ((3) in Section 3.1). Clearly, the APs operate independently and the whole network does not need any synchronization.

TABLE 1
Simulation Settings

Medium Access Protocol	IEEE 802.11b
Radio Propagation Model	Shadowing
Link Basic Rate	2 Mbps
Link Data Rate	11 Mbps
RTS/CTS	OFF
Bounding box size	500m by 500m
Maximum Clients for an AP	8
Transmission Power	15 dBm
Packet Size	1000 bytes
Simulation Trials	10

Algorithm 1 CACAO Algorithm

```
CACAO($ap_i$)
 1. Initialization - Initial Assignment
      $ap_i.c \leftarrow rand(k)$
 2. Optimization - Repeated for each AP
      2.a $GatherStatistic()$
      2.b $c_t = ComputeInterference()$
      2.c $SwitchTo(c_t)$
```

$GatherStatistic()$ is a procedure that is used to gather statistics from clients. It returns the traffic information collected by clients (see Table 1 in supplementary file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2011.59). Then the AP's optimization routine performs computations and comparisons to decide which channel the AP will use for the next time period by procedure $ComputeInterfere()$. This routine computes the expected interference level that the entire BSS will experience for the next time interval. The channel with the least expected interference level will be chosen. Finally the $SwitchTo()$ routine assigns the AP and its associated clients to the chosen channel.

Next, we prove that this distributed algorithm converges for a network with stable topology and data rate. Whenever an AP performs the optimization, the graph is divided into two parts. One is the local graph $G(ap_i)$. The other one is G', which is the original graph G minus ap_i. For $G(ap_i)$, the AP's channel switching decision guarantees that the sum of interference level of $G(ap_i)$ decreases. For G', since ap_i is taken out, the sum of interference level remains the same. Therefore, a channel switching action guarantees that the

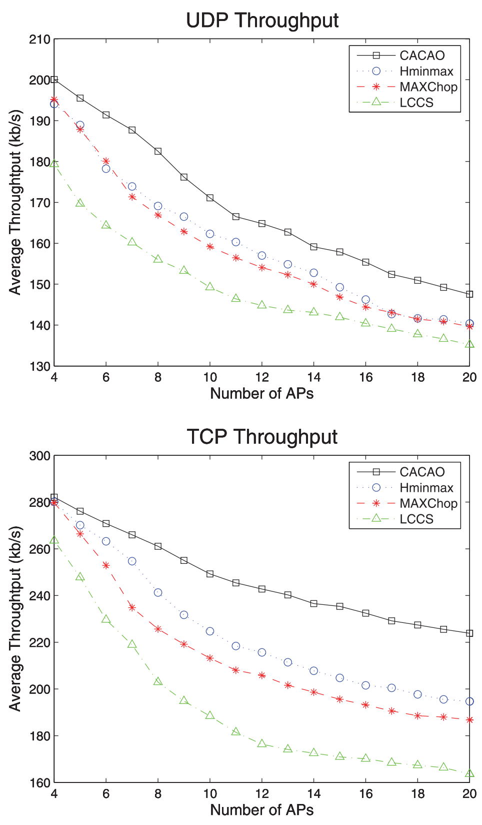


Fig. 5. Average throughput per flow against number of APs for different schemes.

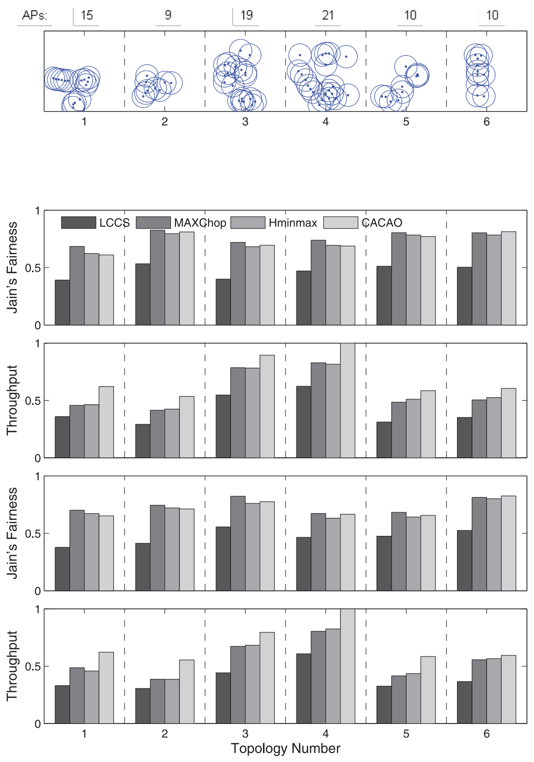


Fig. 6. Simulation results over six topologies in the urban area, derived from wireless network database $Wigle$. The bars represent, from left to right, LCCS, MAXChop, Hminmax, and CACAO.

total interference level for graph G decrease. Therefore, the global interference level is also guaranteed to converge to some (local) minimum point over time.

4 Illustrative Simulation Results

4.1 Simulation Metrics and Environment

We evaluate CACAO using packet-level simulations with NS-2 network simulator. We compare the performance of CACAO with three other channel assignment algorithms, namely, the widely implemented LCCS, MAXChop, and Hminmax.

We are mainly interested in the following performance metrics: 1) Aggregate network throughput, which allows us to study the impact of channel assignment algorithms on the overall network throughput; and 2) Fairness of per-AP throughput, which is an important metric for independent WLANs. We measure the fairness in per-AP throughput using Jain's fairness index given by

$$F(x_1, x_2, \dots, x_n) = \frac{\left[\sum_{i=1}^{n} x_i\right]^2}{n \sum_{i=1}^{n} {x_i}^2}, \tag{5}$$

where the $x's$ represent the throughput of the APs.

We conduct our simulations on some deployed and simulated topologies. Unless otherwise stated, we use the values specified in Table 1 in our simulations. For simulated topologies, we generate random networks. First, we randomly place APs inside a rectangle bounding box. For each AP, the number of associated clients is randomly drawn from one to some maximum value. After the number of associated clients is determined for each AP, we randomly put them inside the coverage area of their associated AP. For simulations using UDP traffic, we generate CBR traffic with constant packet size using the value specified in the table. For simulations using TCP traffic, an FTP application is created, and no data rate is specified. RTS/CTS is turned off because it is the default setting in most commercial APs [21].

4.2 Illustrative Results

We compare the average throughput per flow of CACAO, Hminmax, MAXChop, and LCCS in Fig. 5. For all algorithms, the average user throughput decreases with the increase of the number of independent WLANs due to increasing interference and contention. When the number of WLANs is small, the throughput achieved by all algorithms is roughly the same (due to low interference). As the number of WLANs increases, the performance of LCCS decreases dramatically because it uses only AP side static channel assignment. The performance of Hminmax and MAXChop are much better. CACAO outperforms all three by considering channel condition information to reduce interference. Fig. 5 shows that CACAO achieves the highest throughput for both UDP and TCP connections. It shows that the channel condition information gathered by clients is important for the APs to make good channel decisions to minimize interference between independent WLANs.

We next study the performance of CACAO and other algorithms on some real and representative topologies, derived from the popular wireless network database $Wigle$ [22]. Six different topologies are used, as shown in Fig. 6. AP placements and interference area are shown on the graph as

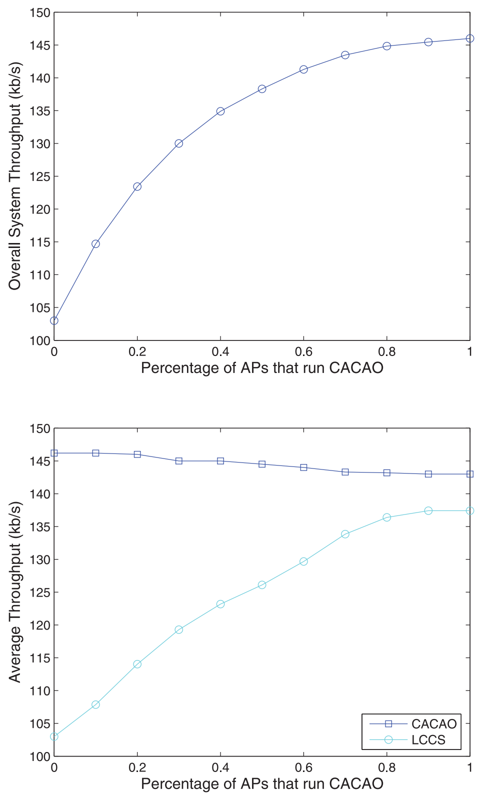


Fig. 7. Overall system throughput improvements and performance measurements for individual algorithms.

dots and circles, respectively. Clients are located randomly inside the transmission range of their associated APs.

Simulation results for all topologies are shown in Fig. 6. For each topology, throughput and fairness for both UDP and TCP flows are compared using four channel assignment algorithms. In all topologies, CACAO achieves the highest throughput due to its better channel selection. The fairness indexes of LCCS are the lowest, because some of the APs have very low throughput due to improper channel allocation. Fairness index achieved by CACAO, MAXCHOP, and Hminmax algorithm are similar. The high fairness index of MAXCHOP is due to its channel hopping. Since there are many slots during one hopping sequence, an AP may experience high interference in some slots and low interference in other slots. As a result, the overall throughput of each AP is similar. On the other hand, CACAO algorithm addresses the fairness problem from a different angle. Unfairness comes mainly from the fact that some areas have high interference and low throughput. Therefore, by minimizing the total interference, systems using CACAO can achieve more uniformly distributed throughput (as to each AP) and better fairness index.

Next, we explore compatibility issue of CACAO by studying the performance gain of incremental deployment of CACAO in a normal network that uses LCCS. In Fig. 7a, we show the overall system throughput improvement versus the percentage of APs running CACAO. The performance improves quickly in the beginning and then flattens at some value (around 45 percent). This is because when most of the APs run LCCS, channel utilization is not even. Therefore, APs that run CACAO have more opportunities to switch to underutilized channels. This can directly increase the overall throughput by reducing interference and can increase overall performance quickly. Fig. 7b shows the TCP throughput for APs that run CACAO and LCCS separately, in the same simulation environment. As shown in this figure, APs switching to CACAO not only improve their own performance, but also boost the performance of APs that still run LCCS.

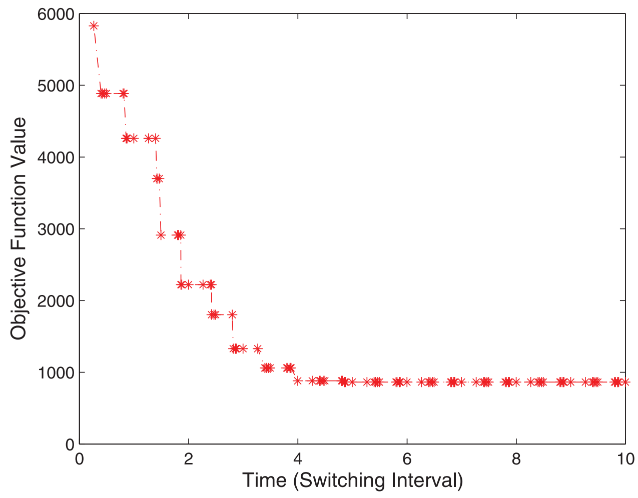


Fig. 8. Objective function value over time on the fourth sample topology shown in the topology map (21 APs). Constant CBR traffic is used.

We show CACAO's convergence in Fig. 8 by presenting a typical evolution of the global objective function over time (for the fourth topology in the topology map with 21 APs). CBR traffic is generated for this study and does not change over time. As shown in the figure, the objective function value decreases dramatically because APs switch to channels with less traffic and avoid interference. CACAO converges to a stable point after only a few rounds of optimization (an average of 5.4 rounds for all six sample topologies).

We compare CACAO's throughput with the optimal solution based on brute force method in Fig. 9. Due to the NP-hard nature of the problem, the performance comparison is done using a small network. Obviously, the brute force algorithm performs better. However, it is a centralized algorithm with exponential running-time complexity. CACAO achieves high throughput with small performance penalty (within 20 percent).

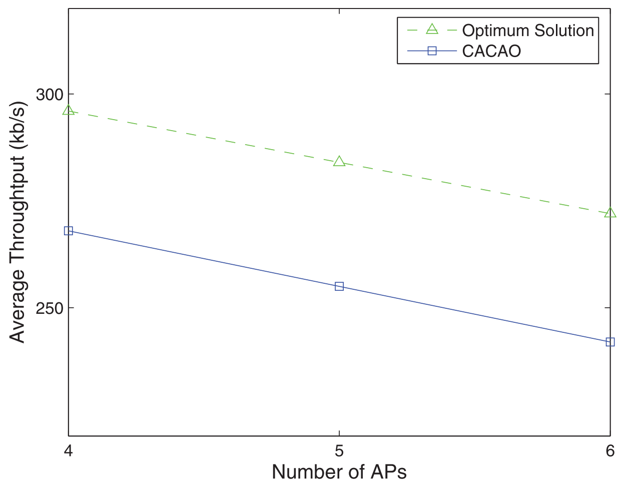


Fig. 9. Throughput comparison between CACAO and Global Optimum Solution.

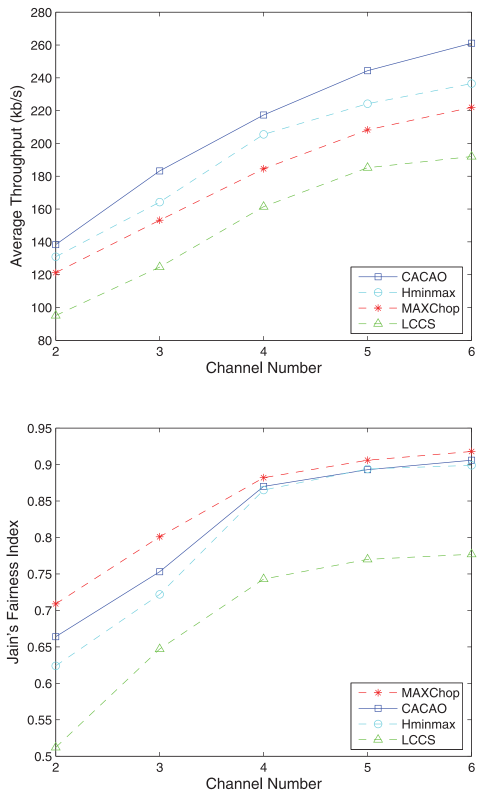


Fig. 10. Throughput and Jain's fairness index for the fourth sample topology shown in the topology map (21 APs). Time varying TCP flows are used in these simulations.

We show the normalized throughput per flow and fairness index of the system versus the number of channels in Fig. 10. Clearly, the increase of nonoverlapping channels has a positive impact on both per flow throughput and fairness index. Overall, CACAO achieves the highest throughput and a fairness index very close to MAXChop.

5 Conclusion

In recent years, WLANs have became more and more popular due to the pervasiveness of wireless devices. Many of these WLANs are independently set up by uncoordinated and inexperienced users. Therefore, developing an automatic and efficient channel assignment algorithm becomes very important to these uncoordinated WLANs.

In this paper, we formulate the channel assignment problem in uncoordinated WLANs as an optimization problem. After proving that it is NP-hard, we propose a distributed algorithm CACAO that tries to minimize the interference level in the network.

CACAO overcomes the weaknesses of the commonly used LCCS and other approaches. It is simple, effective, completely distributed, and hence scalable. APs using CACAO gather channel interference conditions statistics with the help of their associated clients. This information helps the APs to make better decisions on channel assignment to reduce interference.

Using NS2, we implement the traditional LCCS, recently developed MAXChop algorithm and Hminmax algorithm, and our CACAO algorithm and run extensive simulations to study their performance. Our results show that CACAO improves network throughput significantly with little compromise in fairness. It converges fast, and is adaptive to traffic conditions to keep interference at a low level.

References

[1] Nintendo Wi-Fi Connection website, http://www.nintendowifi.com/, 2011.
[2] PlayStation Portable website, http://www.us.playstation.com/, 2011.
[3] The Wi-Fi Alliance, http://wi-fi.org/, 2011.
[4] ABI Research, http://www.abiresearch.com/, 2011.
[5] A. Akella, G. Judd, S. Seshan, and P. Steenkiste, "Self-Management in Chaotic Wireless Deployments," *Proc. ACM MobiCom*, 2005.
[6] K. Jain, J. Padhye, V.N. Padmanabhan, and L. Qiu, "Impact of Interference on Multi-Hop Wireless Network Performance," *Proc. ACM MobiCom*, 2003.
[7] C. Hua and R. Zheng, "Starvation Modeling and Identification in Dense 802.11 Wireless Community Networks," *Proc. IEEE INFOCOM*, 2008.
[8] "IEEE 802.11k—Radio Resource Measurement Enhancements," http://grouper.ieee.org/groups/802/11/Reports/tgk_update.htm, 2011.
[9] A. Mishra, V. Shrivastava, D. Agrawal, S. Banerjee, and S. Ganguly, "Distributed Channel Management in Uncoordinated Wireless Environments," *Proc. ACM MobiCom*, 2006.
[10] A. Mishra, S. Banerjee, and W. Arbaugh, "Weighted Coloring Based Channel Assignment for WLANs," *Proc. ACM SIGMOBILE Mobile Computing and Comm. Rev.*, 2005.
[11] A. Hills, "Large-Scale Wireless Lan Design," *IEEE Comm. Magazine*, vol. 39, no. 11, pp. 98-107, Nov. 2001.
[12] K.K. Leung and B.-J.J. Kim, "Frequency Assignment for IEEE 802.11 Wireless Networks," *Proc. IEEE Vehicular Technology Conf.*, 2003.
[13] Y. Lee, K. Kim, and Y. Choi, "Optimization of AP Placement and Channel Assignment in Wireless LANs," *Proc. IEEE Conf. Local Computer Networks*, 2002.
[14] I. Broustis, K. Papagiannaki, S.V. Krishnamurthy, M. Faloutsos, and V. Mhatre, "Mdg: Measurement-Driven Guidelines for 802.11 WLAN Design," *Proc. ACM MobiCom*, 2007.
[15] R. Murty, J. Padhye, R. Chandra, A. Wolman, and B. Zill, "Designing High Performance Enterprise Wi-Fi Networks," *Proc. Symp. Networked Systems Design and Implementation (NSDI)*, 2008.
[16] A. Mishra, V. Brik, S. Banerjee, A. Srinivasan, and W. Arbaugh, "A Client-Driven Approach for Channel Management in Wireless LANs," *Proc. IEEE INFOCOM*, 2006.
[17] N. Ahmed and S. Keshav, "A Successive Refinement Approach to Wireless Infrastructure Network Deployment," *Proc. IEEE Wireless Comm. and Networking Conf.*, 2006.
[18] B. Kauffmann, F. Baccelli, A. Chaintreau, V. Mhatre, K. Papagiannaki, and C. Diot, "Measurement-Based Self Organization of Interfering 802.11 Wireless Access Networks," *Proc. IEEE INFOCOM*, 2007.
[19] E. Rozner, Y. Mehta, A. Akella, and L. Qiu, "Traffic-Aware Channel Assignment in Wireless LANs," *Proc. ACM SIGMOBILE Mobile Computing and Comm. Rev.*, 2007.
[20] E. Rozner, Y. Mehta, A. Akella, and L. Qiu, "Traffic-Aware Channel Assignment in Enterprise Wireless Networks," *Proc. IEEE Int'l Conf. Network Protocols*, 2007.
[21] J. Robinsony, K. Papagiannakiz, C. Diotz, X. Guo, and L. Krishnamurthy, "Experimenting with a Multi-Radio Mesh Networking Testbed," *Proc. First Workshop Wireless Network Measurements*, 2005.
[22] "Wigle: Wireless Geographic Logging Engine," http://www.wigle.net/, 2011.

Xiaonan Yue received the BEng in computer engineering and an MPhil degree in computer science from The Hong Kong University of Science and Technology, Hong Kong, in 2007 and 2010, respectively. He is currently an analyst programmer at J.P. Morgan, Hong Kong. His research interests are on wireless networks and multimedia networking.

Chi-Fai Michael Wong received the BEng in computer engineering and an MPhil in computer science from The Hong Kong University of Science and Technology, Hong Kong, in 2005 and 2007, respectively. His research interests are on wireless networks and multimedia networking.

Shueng-Han Gary Chan is currently an associate professor at the Department of Computer Science and Engineering in The Hong Kong University of Science and Technology, Hong Kong, and an adjunct researcher in the wireless and networking group at the Microsoft Research, Asia. He was a visiting assistant professor at the Department of Computer Science, University of California, Davis, from 1998 to 1999. He was a William and Leila fellow at Stanford University, where he did his PhD in electrical engineering and a minor in business administration (with a concentration in management of operations, information and technology). His PhD thesis, supervised by professor Fouad Tobagi, was on the provisioning of scalable distributed video services.