International Journal of Networking and Computing – www.ijnc.org
ISSN 2185-2839 (print) ISSN 2185-2847 (online)
Volume 5, Number 1, pages 200–222, January 2015

Extensions of Access-Point Aggregation Algorithm for Large-scale Wireless Local Area Networks

Md. Ezharul Islam, Nobuo Funabiki
Department of Electrical and Communication Engineering, Okayama University
3-1-1 Tsushimanaka, Okayama 700-8530, Japan

Toru Nakanishi
Department of Information Engineering, Hiroshima University
1-4-1 Kagamiyama, Higashi-Hiroshima, 739-8527, Japan

Received: February 14, 2014
Revised: May 3, 2014
Revised: July 20, 2014
Accepted: August 2, 2014
Communicated by Yoshiaki Kakuda

Abstract

Recently, many organizations such as universities and companies have deployed *wireless local area networks (WLANs)* to cover the whole site for ubiquitous network services. In these WLANs, wireless *access-points (APs)* are often managed independently by different groups such as laboratories or departments. Then, a host may detect signals from multiple APs, which can degrade the communication performance due to radio interferences among them and increase operational costs. Previously, we proposed the *AP aggregation algorithm* to solve this problem by minimizing the number of active APs through aggregating them using the *virtual AP technology*. However, our extensive simulations in various instances found that 1) the minimization of active APs sometimes excessively degrades the network performance, and 2) the sequential optimization of host associations does not always reach optimal where slow links are still used. In this paper, we propose *two extensions* of the AP aggregation algorithm to solve these problems by 1) ensuring the *minimum average throughput* for any host by adding active APs and 2) further optimizing host associations by changing multiple hosts simultaneously in the *host association finalization* phase. We verify the effectiveness through simulations in four network instances using the WIMNET simulator.

Keywords: Wireless local area network, access-point aggregation, minimum host throughput, host association, heuristic algorithm, IEEE 802.11n

1 Introduction

Currently, a *wireless local area network (WLAN)* [1, 2] has become a popular choice in many organizations such as universities and companies for communications among their employees and the Internet accesses. *Access-points (APs)* in these WLANs are often independently managed by different groups such as departments or laboratories in organizations even in the same building. As a result, their coverage areas can be overlapped, and a WLAN user host can detect signals from multiple APs. This situation may degrade the performance of WLANs by increasing interferences among them, because the number of non-interfered frequency channels in the *IEEE802.11 WLAN protocol* is limited. They also increase unnecessary energy consumptions resulting in increasing operational costs [3].

To solve this problem, we have studied the aggregation of installed physical APs into a smaller number of active APs using the *virtual AP (VAP) technology* [4–6]. Each virtual AP can be configured independently by assigning an individual SSID and security options with the negligible performance overhead even on the same physical AP [7]. We formulated the *AP aggregation problem* for multiple WLANs to minimize the number of active physical APs under the grouping and minimum link speed constraints, and proposed its heuristic algorithm [8].

However, our extensive simulations found two problems in the previous algorithm. The first problem is that the minimization of active APs may excessively degrade the performance of WLANs. A smaller number of active APs increases the number of associated hosts with each active AP, and reduces the throughput for each host. The second problem is that sequential changes of host associations cannot always reach optimal where slow links are still used. Simultaneous changes of multiple host associations may be sometimes necessary to escape from a deep local minimum.

In this paper, we propose two extensions of the AP aggregation algorithm to solve these problems. The first extension is to ensure the *minimum average throughput* for any host in the field by adding active APs, after minimizing the number of active APs. This paper shows that the average throughput can be estimated using the cost function that has been defined in the algorithm. Extra APs are activated when the estimated average throughput does not reach the given threshold. The second extension is to apply the *host association finalization* phase of changing multiple host associations simultaneously after sequential changes. To verify their effectiveness, we apply the extended algorithm to four network instances, where throughputs by their solutions are evaluated using the WIMNET simulator [9].

Within our knowledge, the same AP aggregation problem of this paper has not been reported. However, several related works have been reported in literature.

In [10], Jardosh et al. studied the energy loss scenario due to idle WLAN resources and proposed a resource on-demand (RoD) based strategy to minimize the power consumption by powering on/off APs dynamically, based on the volume and locations of user demands. In [11], Ganji et al. also studied the energy efficiency issue in dense WLANs and proposed an aggressive scheme for adaptation of the AP density to actual traffics. This method keeps APs inactive to the extent so that the remaining active APs can provide the coverage required to detect the host connectivity initiation. Once a host is detected, additional APs are activated to provide the required service. These approaches basically offer the mechanism of adaptively changing active APs for traffic demands from hosts when they are changed. On the other hand, our approach can optimize active APs and host associations for a given static traffic demand to reduce energy consumptions while maximizing the performance. Unfortunately, our approach does not consider the dynamic change of demands. Thus, the combination of their approaches and ours is significant to achieve both the optimality and the dynamics, which will be in our future works.

In [12], Ling et al. proposed a local search algorithm for the joint AP placement and channel assignment for 802.11 WLANs. It can provide the better performance with the lower time complexity than the exhaustive search. In [13], Eisenblatter et al. also proposed a joint optimization algorithm for the AP placement and the channel assignment using a mathematical programming. They show the superiority of this integrated approach to the conventional sequential one. On the other hand, our approach considers predefined positions of the APs, and reduces the number of active APs while maximizing the performance.

In [14], Funabiki et al. proposed an active AP selection algorithm for a wireless mesh network. On the other hand, our approach considers a general wired-backbone WLAN.

In [15], Xu et al. proposed a decentralized AP selection algorithm to achieve a minimum throughput with the reasonable computation and transmission overhead to be compatible with legacy APs. In [16], Miyata et al. also proposed an AP selection algorithm to maximize the throughput while preserving newly arrived-user throughputs in a multi-rate WLAN. It supports limited user movements. Their goal is to maximize the throughput by optimizing the associated APs for hosts while the locations of the APs are fixed. On the other hand, we focus on reducing the number of active

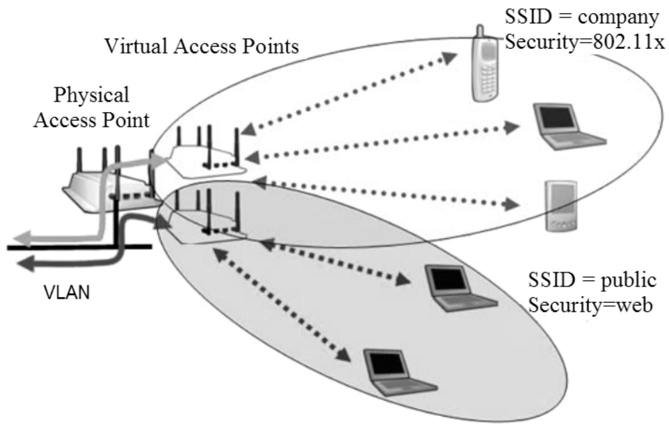

Figure 1: One physical AP with two virtual APs.

APs while maximizing the performance by optimizing the associated APs for hosts.

The rest of the paper is organized as follows: Section 2 provides the background technologies for this study. Section 3 presents the extensions of AP aggregation algorithm. Section 4 presents the extended algorithm procedure. Section 5 discusses simulation results for evaluations. Section 6 concludes this paper with some future works.

2 Background Technologies

In this section, we overview the virtual AP technology and the IEEE 802.11n protocol.

2.1 Virtual Access Point

A *virtual access point (VAP)* is a technology to allow a single physical AP to virtually serve as multiple logical APs using virtualization technique. Currently, most commercial AP devices support this VAP technology. Each VAP can be configured independently by assigning an individual SSID and security options, and appears to the hosts as an independent physical AP as shown in Figure 1.

2.2 IEEE 802.11n Protocol

Our AP aggregation approach assumes the use of the IEEE 802.11n protocol [17] to enhance the performance of WLANs. Here, we briefly introduce adopted technologies and the feature of the link speed change by the distance.

2.2.1 Characteristic Technologies for High Speed Communication

The IEEE 802.11n protocol adopts advanced communication technologies to achieve higher throughputs for wireless communications [18]. They include the *channel bonding*, the *MIMO channel* with the *spatial division multiplexing* and the *space-time block coding*, the *short guard interval*, and the *frame aggregation*.

In the *channel bonding*, one channel can double the physical data rate by using two adjacent $20MHz$ channels simultaneously. In the *MIMO channel*, one channel can increase the data rate by using multiple antennas. In the *spatial division multiplexing*, a sender can transmit different data streams simultaneously along multiple antennas. In the *space-time block coding*, a sender can transmit one data stream over multiple antennas, which can also increase the interference tolerance. In the *short guard interval*, the guard interval is reduced from 800 ns to 400 ns. In the *frame*

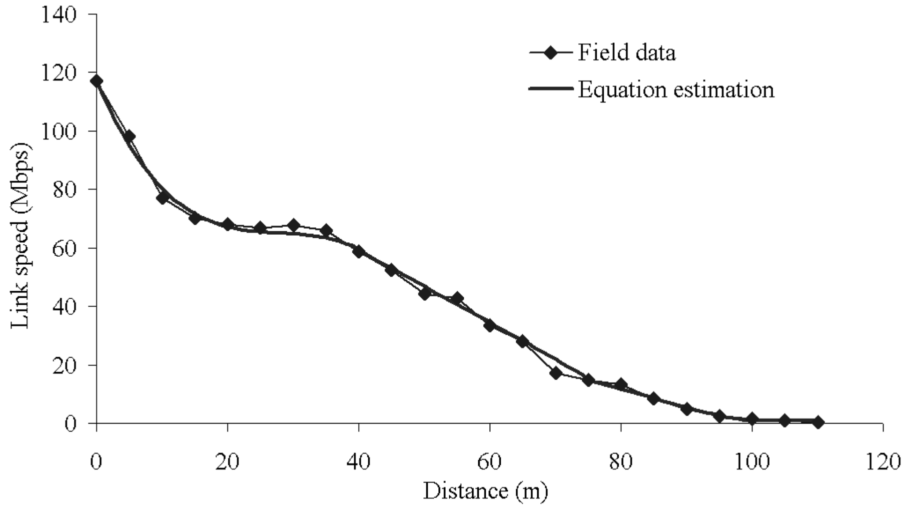

Figure 2: Link speed change for IEEE802.11n.

aggregation, multiple frames can be transmitted as one aggregated frame using *MPDU (MAC protocol data unit)* to reduce the overhead of medium access.

2.2.2 Link Speed Change Feature

In our preliminary experiments [19], we measured the link speed when both end nodes adopted the IEEE 802.11n protocol and the distance between them is increased from $1m$ to $110m$ with the $5m$ interval in a free space. As shown in Figure 2, the link speed drastically decreases as the distance increases.

Then, we approximated this link speed change $f(x)$ by the third-order equation of the link distance x as follows:

$$f(x) = \begin{cases} -0.0022x^3 + 0.1853x^2 - 5.3348x + 117.43 & (0 < x < 40) \\ -0.00006x^3 + 0.0095x^2 - 1.732x + 117.17 & (40 \leq x < 75) \\ 0.000438x^3 - 0.10955x^2 + 8.477156x - 189.481818 & (75 \leq x < 100) \\ 1.0 & (100 \leq x) \end{cases} \tag{1}$$

where x represents the link distance (m), and $f(x)$ represents the estimated link speed (throughput) at distance x ($Mbps$) from the source node. The estimated result using this equation is also plot by the thick line in Figure 2.

The link speed between an AP and a host should be measured using wireless devices in a real network field if possible. When it is impossible or hard, the link speed should be estimated using $f(x)$ in Equation (1) from the locations of the APs and the hosts. Then, the link speed sp_{ij} between AP_i and $host_j$ can be estimated from the Euclid distance by:

$$sp_{ij} = f\left(\sqrt{(x_i^{AP} - x_j^{host})^2 + (y_i^{AP} - y_j^{host})^2}\right) \tag{2}$$

where (x_i^{AP}, y_i^{AP}) and (x_j^{host}, y_j^{host}) represent the location coordinate of AP_i and that of $host_j$ respectively.

3 Extensions of AP Aggregation Algorithm

In this section, we present the two extensions of the AP aggregation algorithm.

3.1 Summary of Extensions

Our simulations of the previous AP aggregation algorithm found the following problems in the performance:

1. The minimization of the number of active APs sometimes excessively degrades the performance of WLANs.

2. The host association cannot always reach optimal due to the sequential optimization process.

To solve these problems, we propose the following two extensions of the AP aggregation algorithm in this paper:

1. The minimum average throughput for any host in the network field is ensured by adding active APs after their minimization.

2. The *host association finalization* phase is added to optimize host associations by changing multiple host associations simultaneously.

3.2 First Extension: Ensuring Minimum Average Throughput

When the number of active APs is minimized, the number of hosts associated to one AP increases, which reduces the communication performance per host. The performance per host is actually very important for each user in WLAN. Because the individual performance cannot be evaluated by the algorithm, we use the estimated average throughput per host that can be calculated from our defined cost function. Then, we introduce the *average throughput constraint* that the estimated *minimum average throughput* per host must satisfy the given minimum throughput G called the *average throughput threshold*. The number of active APs may be increased when this constraint is not satisfied.

When an AP (let AP_i) is associated with a host (let $host_j$) with the link speed (let sp_{ij}), the required time to transmit one bit is $t_{ij} = 1/sp_{ij}$. If multiple hosts are associated with AP_i, the total time becomes $\sum_{j \in AH_i} t_{ij}$, where AH_i represents the set of such hosts. Because each host can transmit one bit during this time if every host transmits the same amount of data and their transmission chances are fair, the *minimum average throughput* per host (let $minT$) can be estimated by:

$$\begin{aligned} minT &= min(\frac{1}{\sum_{j \in AH_i} t_{ij}}) \\ &= min(\frac{1}{\sum_{j \in AH_i} \frac{1}{sp_{ij}}}) \\ &= \frac{1}{max(T_i)} = \frac{1}{E_2} \end{aligned} \tag{3}$$

where T_i is the transmission time of AP_i, and E_2 is the maximum transmission time among the active APs that has been defined as a cost function in the previous algorithm [8].

Thus, from Equation (3), to satisfy the *average throughput constraint*, the inverse of E_2 must be G or larger:

$$\frac{1}{E_2} \geq G. \tag{4}$$

3.3 Second Extension: Host Association Finalization

The sequential optimization of host associations in the previous algorithm cannot always reach optimal due to the heuristic nature. In this paper, we add the *Host association finalization* phase to further optimize host associations by changing multiple host associations simultaneously. Actually, in this phase, the associated APs of two hosts are swapped if the resulting cost function is improved.

3.4 Extended Problem Formulation

The AP aggregation problem formulation is extended to deal with the two extensions as follows:

3.4.1 Inputs

- Number of physical APs: N
- Number of user hosts: M
- Number of groups: L
- Group pair cooperation flag: a_{ij} $(i, j = 1 \sim L)$
  - $a_{ij} = 1$: groups i and j can be cooperated.
  - $a_{ij} = 0$: groups i and j cannot be cooperated.
- Link speed threshold: H
- Average throughput threshold: G
- Information for AP_i $(i = 1 \sim N)$:
  - Group ID: $1 \sim L$
  - Set of associable hosts.
  - Link speed sp_{ij} for each associable $host_j$, or location coordinate (x_i^{AP}, y_i^{AP})
- Information for $host_i$ $(i = 1 \sim M)$:
  - Group ID: $1 \sim L$
  - Location coordinate: (x_i^{host}, y_i^{host})

3.4.2 Outputs

- Set of active physical APs.
- Set of group IDs assigned to each active AP for virtual APs.
- Associated AP to each host.

3.4.3 Constraints

- Only APs in cooperative groups must be aggregated (group constraint).
- Any host must be associated with a VAP in the same group (host cover constraint).
- Any link between a host and its associated AP must satisfy the minimum *link speed threshold* H (link speed constraint).
- Any AP must satisfy the *average throughput threshold* G (average throughput constraint).

3.4.4 Objective

- To minimize the number of active physical APs ($= E_1$) while satisfying the *average throughput constraint*.
- Holding the first objective, to maximize throughput by minimizing the maximum transmission time among the active APs ($= E_2$).

4 Extended Algorithm Procedure

In this section, we describe the algorithm procedure by extending the one in [8] to afford the two extensions in the previous section.

4.1 Summary of Modifications

Here, we summarize the modifications in the previous AP aggregation algorithm to deal with the two extensions:

1. First extension: ensuring minimum average throughput
   The *AP selection optimization* phase is modified to consider the *average throughput constraint*.
   - A new variable P is introduced to represent the required number of active APs to satisfy the *average throughput constraint*, where P is initialized by 0 in 1).
   - In 2)-(1)-(d), if the current number of active APs is smaller than P, activate randomly selected non-active APs until this number becomes P.
   - A new step 3) is introduced to check the *average throughput constraint* and the termination of the algorithm after the AP selection optimization repetition in 2)-(1). Here, if the improvement of $1/E_2$ is smaller than a newly introduced *throughput improvement margin* V $(Mbps)$, the algorithm is terminated because the average throughput cannot be improved. Otherwise, if the constraint is not satisfied, P is incremented by 1 and go back to the AP selection optimization repetition.
2. Second extension: host association finalization
   In the *host association finalization* phase, the associated APs of two hosts are swapped if the resulting new cost function E_3 is improved. E_3 represents the total transmission time by:

$$E_3 = \sum_{i \in \{1,\ldots,n\}} T_i. \tag{5}$$

   Here, n is the number of active APs, and T_i is the transmission time of AP_i.

4.2 First Phase: Initialization

This phase initializes the variables for the following phases including the link speed estimation and the list generation of the associable hosts for each AP.

1) Estimate the speed of the link between an AP and its associable host by using the method in Section 2.2.2 if necessary.

2) For each physical AP, (i) extract a host that can be associated with it by considering the *group pair cooperation flag* and the *link speed threshold* to generate a list of its associable hosts (*associable host list*) to the AP, and (ii) calculate the number of associable hosts. If any host cannot satisfy the *link speed threshold* for any AP, associate such a host with the AP that can satisfy the group cooperation constraint and provide the maximum link speed.

3) Initialize every physical AP as a *non-active AP*.

4.3 Second Phase: AP Selection

This phase selects active physical APs and their associated hosts to minimize the number of active physical APs (E_1) by the greedy method [20] of repeating the following procedure until every host is associated with an active AP.

1) Select one physical AP as an *active AP* from *non-active APs*, such that the sum of the link speeds with its associable hosts in *associable host list* is maximum among all *non-active APs*.

2) Associate any associable host to this selected AP and count the number of associated hosts with this AP.

3) Remove every host associated in 2) from the *associable host list*, and update the number of associable hosts for any *non-active AP*.

4.4 Third Phase: Host Association Optimization

This phase modifies the AP-host association to improve the overall throughput by minimizing the maximum transmission time among the active APs (E_2) by a local search method [20]. Three algorithm parameters R, S, and Q are used here, which represent the *hill-climbing repetition factor for association*, the *local search repetition factor for association*, and the *association mutation ratio* respectively.

1) (i) Re-associate any host with the associable *active AP* that provides the maximum link speed among the *active APs*, (ii) keep this state as the *best-found association*, (iii) calculate the initial cost function E_2, and (iv) keep it as E_2^{best} (best-found E_2).

2) Repeat the following steps with R times, where R is the *hill-climbing repetition factor for association*.

    (1) Identify the AP with maximum transmission time and make a list of movable hosts associated with this AP, where a host can be usually associated with two or more active APs.

    (2) Repeat the following step with S times or until no movable host is found for the AP with maximum transmission time, where S is the *local search repetition factor for association*.

        (a) (i) Randomly select a movable host from the *associable host list*, (ii) randomly select a new associable *active AP* for the selected host, (iii) associate the selected host to the selected AP, and (iv) calculate the new cost function E_2'.

        (b) (i) Calculate $\Delta E_2 = E_2' - E_2$. (ii) If $\Delta E_2 \leq 0$, set $E_2 = E_2'$, stay on the new state, newly identify the AP with the maximum transmission time, and make a new list of movable hosts associated with this AP. (iii) Otherwise, go back to the previous state.

    (3) If the new cost function E_2 is smaller than E_2^{best}, update E_2^{best} with E_2 and the *best found association* with the current association state.

    (4) Re-associate any host with the associable *active AP* that provides the maximum link speed among the *active APs*.

    (5) Randomly select ($Q \times$ *total number of movable hosts*) movable hosts and randomly change their associated APs where Q is the *association mutation ratio*.

3) Return the *best found association*.

4.5 Fourth Phase: AP Selection Optimization

This phase optimizes the selection of *active APs* with AP-host associations to further minimize both E_1 and E_2 by another local search method while satisfying the average throughput threshold G. If the average throughput constraint is not satisfied by the current number of active APs, this phase increases it. Three algorithm parameters U, T, and V are used here, which represent the *AP selection optimization repetition factor*, the *AP selection optimization ratio*, and the *throughput improvement margin* respectively.

1) (i) Initialize O^{best}, E_1^{best}, and E_2^{best} by the current algorithm output and values of E_1 and E_2 respectively, (ii) initialize the *selection flag* for every AP by OFF, and (iii) initialize *minimum number of APs needed*, $P = 0$, where O^{best}, E_1^{best}, and E_2^{best} represents the best-found output of the algorithm and its values of E_1 and E_2 at O^{best}, respectively.

2) Repeat the following steps with U times, where U is the *AP selection optimization repetition factor*.

    (1) Repeat the following steps with $T \times E_1^{best}$ times, where T is the *AP selection optimization ratio*.

(a) (i) Randomly select one *active AP* with the OFF selection flag, (ii) deactivate it, and (iii) set the selection flag ON for this AP.

(b) (i) Find a new associable *active AP* for each host associated with the AP deactivated in (a), and (ii) associate the host with the AP, where if there are multiple *active APs* associable for the host, select the one that is associated with the largest number of hosts among them.

(c) If some hosts still cannot be associated, repeat the following steps until every host can be associated with an *active AP*:
(c-1) Find a *non-active AP* with the OFF selection flag that can be associated with the largest number of such hosts (hosts with no associated APs).
(c-2) If there is no such *non-active AP*, find a *non-active AP* with the ON selection flag that can be associated with the largest number of such hosts.
(c-3) In both cases, if multiple APs are found, select the *non-active AP* that provides the largest total link speeds to the hosts for tie break.
(c-4) (i) Select this AP as an *active AP*, (ii) associate the unassociated hosts associable to this AP, and (iii) set the selection flag ON for this AP.

(d) If the new number of *active APs* ($= E_1$) is smaller than P, apply the following steps until it reaches P:
(d-1) Randomly select a *non-active AP* with the OFF selection flag. If there is no such *non-active AP*, randomly select a *non-active AP* with the ON selection flag.
(d-2) Select this AP as an *active AP*.

(e) Apply the following steps:
(e-1) Set the selection flag OFF for the newly activated APs in (c).
(e-2) If the new number of *active APs* ($= E_1$) is smaller than E_1^{best}, (i) apply **Third Phase**, (ii) calculate E_2, and (iii) update O^{best}, E_1^{best}, and E_2^{best}.
(e-3) Otherwise, if $E_1 = E_1^{best}$, (i) apply **Third Phase**, (ii) calculate E_2, and (iii) if E_2 is smaller than E_2^{best}, update O^{best} and E_2^{best}.

(f) If at least one *active AP* has the OFF selection flag, go to (a).

(g) Reset the selection flag for every AP by OFF.

(2) If O^{best} is not updated at (1), reset the current state by O^{best}.

3) Apply the following procedure for *average throughput constraint*:

(1) If $(1/E_2^{best}) \geq G$ (*average host throughput threshold*), mark the solution as *success*, and go to 4).

(2) Otherwise, if $E_1^{best} = N$ (*maximum number of APs available*), mark the solution as *maximum number of APs reached*, and go to 4).

(3) Otherwise, if P=0 or $(1/E_2^{best} - 1/E_2^{old}) \geq V$ (*throughput improvement margin*),

(a) Save current O^{best} as O^{old}, E_1^{best} as E_1^{old}, and E_2^{best} as E_2^{old}.

(b) Set $P = E_1^{best} + 1$

(c) (i) Randomly activate a *non-active AP* in O^{best}, (ii) run **Third Phase**, and (iii) update O^{best}, E_1^{best}, and E_2^{best}.

(d) Reset the current state by O^{best} and go back to 2).

(4) Otherwise, reset O^{best} from O^{old} and mark the solution as *saturated*.

4) Return the best-found output O^{best}.

4.6 Fifth Phase: Host Association Finalization

This phase improves and finalizes host association by swapping the association of hosts.

1) Initialize O^{best} and E_2^{best} by the current algorithm output and value E_2 respectively.

Table 1: Algorithm parameter values.

Parameter	Role	Value
R	Hill-climbing repetition factor for association	$5 \times N$
S	Local search repetition factor for association	$40 \times N \times M$
Q	Association mutation ratio	0.1
T	AP-selection optimization ratio	0.5
U	AP-selection optimization repetition factor	N
V	Throughput improvement margin (Mbps)	0.1

2) Select a new pair of hosts that satisfy the following conditions:

   (1) They are not associated with the same AP.

   (2) They belong to the same or cooperable groups.

   (3) This host pair has not been considered this time.

   (4) Each new link satisfies the *link speed constraint* after the swapping.

   (5) The total transmission time E_3 in Equation (5) is decreased after the swapping.

   If no such pair is found, terminate the procedure and return the best-found output O^{best}.

3) Swap their associated APs, and calculate E_2.

4) If $E_2 > E_2^{best}$, resume the previous host associations. Otherwise, update O^{best}, and E_2^{best}.

5) Go to 2).

4.7 Sixth Phase: Virtual AP Assignment

This phase assigns the virtual APs to the active physical APs.

1) For each active AP, assign the VAPs in the same group as the associated hosts if necessary.

4.8 Summary of Algorithm Parameter and Procedure

In our simulations in Section 5, we used the values for the algorithm parameters in Table 1. To improve the readability of the complex procedure and parameters in the proposed algorithm, Figure 3 illustrates the flowchart of the six phases in the algorithm, where R, S, and Q for the third phase are described. Figure 4 illustrates the flowchart for the details of the fourth phase where T, U and V are used.

5 Performance Evaluations by Simulations

In this section, we evaluate the extended AP aggregation algorithm through simulations in four instances using the WIMNET simulator [9].

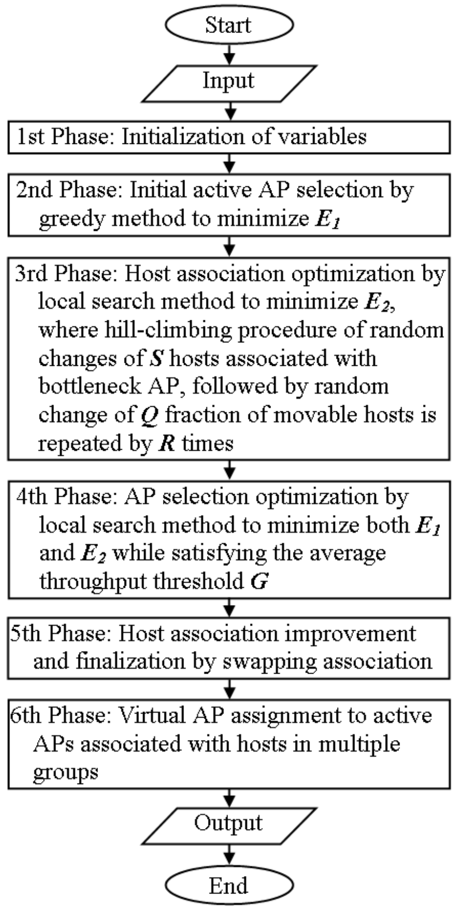

Figure 3: Overview of six-phase AP aggregation algorithm.

Table 2: Simulation environment.

Simulator	WIMNET Simulator
Interface	IEEE 802.11n
CPU	Intel Core 2 Duo 3.33 GHz
Memory	3 GB
Operating System	Ubuntu 12.04 (Linux kernel 3.2.0)

5.1 WIMNET Simulator

The WIMNET simulator simulates least functions for wireless communications of hosts and APs that are required to calculate throughputs and delays. It has originally been developed to evaluate a large-scale *Wireless Internet-access Mesh NETwork (WIMNET)* with reasonable CPU time on a conventional PC. A sequence of functions such as host movements, communication request arrivals, and wireless link activations are synchronized by a single global clock called a time slot. Within integral multiple of time slots, a host or an AP can complete one-frame transmission and the acknowledge reception. Different transmission rate can be set by manipulating the time slot length and the number of time slots for one link activation. For simulations in this paper, we have modified this simulator to simulate WLANs with multiple APs. Table 2 summarizes the hardware and software platforms used for the simulation.

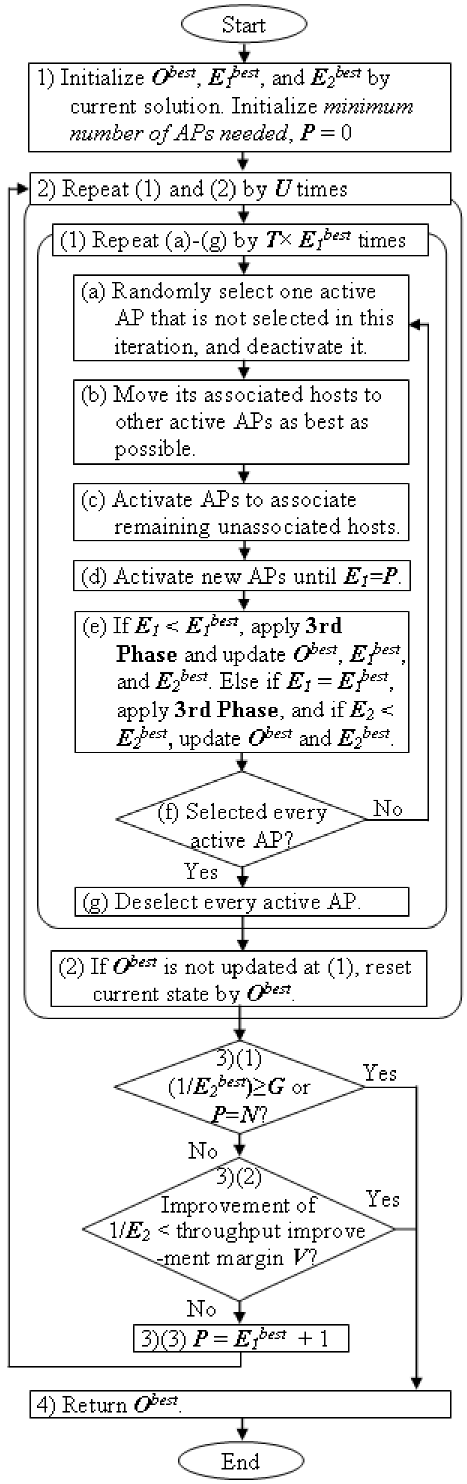

Figure 4: Flowchart for *AP selection optimization* phase.

5.2 Simulation Results for Random Instance

First, we consider the instance with a random topology of 16 APs, 32 hosts, and four groups to verify the effects of the two extensions in the AP aggregation algorithm.

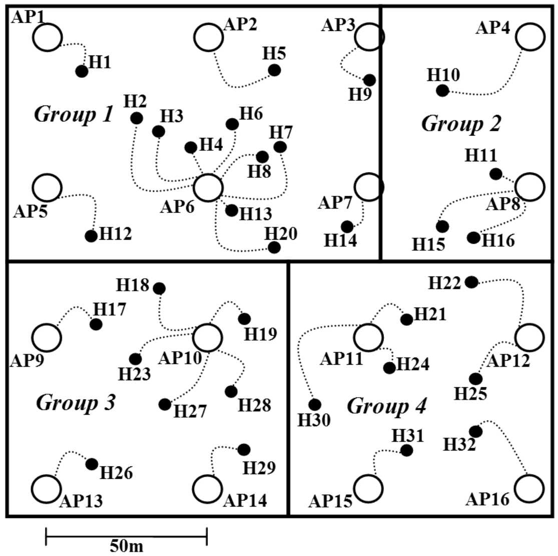

Figure 5: Random instance.

5.2.1 Simulation Instance

Figure 5 illustrates the topology in this *random instance*. **AP1** to **AP16** represent APs, and **H1** to **H32** represent hosts. Any AP is connected to the Internet through a wired cable. An ideal case of an open space is assumed here where each AP has the coverage area with the $110m$ radius. The distance between two neighboring APs is $50m$, and the length of one side of the field is $150m$. A dotted line between an AP and a host represents the initial association with the smallest distance. **AP6** and **AP10** are associated with 8 and 5 hosts respectively, while most APs are associated with only one host. The presence of redundant APs makes this instance useful to show the effectiveness of the AP aggregation algorithm.

5.2.2 Simulation Results

For the random instance, we evaluate the performance of the algorithm when we change the *average throughput threshold* G from $0Mbps$ to $20Mbps$ and the *link speed threshold* H from $20Mbps$ to $60Mbps$. For simplicity, we assign a different channel to any active AP, and consider that all groups can be cooperated. We run the algorithm 10 times for each pair of values of G and H, and take the average values of E_1 and $1/E_2$ of solutions from the algorithm, and of minimum average throughputs and overall throughputs from the WIMNET simulator. Tables 3 and 4 show the results before/after applying the *host association finalization* phase.

5.2.3 Evaluation of Host Association Finalization Phase

The comparison between Tables 3 and 4 shows that, the *host association finalization* phase slightly improves the overall throughput in any G for $H = 20$, and $G = 20$ for $H = 40$ and 60. The average values of $1/E_2$ and the minimum average throughput have also been improved in them.

The search space size for the AP aggregation algorithm increases with the number of available links and the number of active APs in solutions, which is larger for the smaller H and the larger G. Even for such cases, the algorithm can reach better solutions with the *host association finalization* phase and improve the throughout.

To see the effects of the *host association finalization* phase more directly, as typical examples, Figures 6 and 7 show solutions by the algorithm for $G = 0$, 20 and $H = 20$ respectively. The dotted lines represent previous associations before the *host association finalization* phase. In Figure 6, the

Table 3: Simulation results before host association finalization in *random instance*.

Link speed threshold H (Mbps)		Average throughput threshold G (Mbps) 0	10	20
20	Avg. of E_1	4.00	6.00	11.00
	Avg. of $1/E_2$	7.13	10.80	20.77
	Avg. of min. avg. throughput (Mbps)	6.96	10.62	20.52
	Avg. of overall throughput (Mbps)	222.73	339.97	656.65
40	Avg. of E_1	4.00	6.00	11.00
	Avg. of $1/E_2$	7.33	10.88	20.97
	Avg. of min. avg. throughput (Mbps)	7.18	10.70	20.72
	Avg. of overall throughput (Mbps)	229.85	342.28	663.08
60	Avg. of E_1	9.00	9.00	11.00
	Avg. of $1/E_2$	11.55	11.55	21.19
	Avg. of min. avg. throughput (Mbps)	11.38	11.38	20.99
	Avg. of overall throughput (Mbps)	364.04	364.04	671.59

associated APs of **H13**, **H29** and **H20** are rotated, and in Figure 7, the associated APs of host pairs (**H1**, **H3**), (**H4**, **H6**), (**H18**, **H23**) and (**H27**, **H28**) are swapped, which increase the overall throughput respectively.

5.2.4 Evaluation of Minimum Average Throughput

From Table 4, the comparison between the *minimum average throughput* and G indicates that the algorithm can satisfy the *average throughput constraint* for any case, by adding active APs if necessary, when we compare E_1 among different G. The comparison between $1/E_2$ and the *minimum average throughput* shows that the former one in our algorithm is the good estimation of the latter one. However, we should note that if some APs are interfered with each other by using the same channel, the *minimum average throughput* can be lowered than $1/E_2$ due to the interference. The improvement of the minimum average throughput estimation under limited channels is in our future studies.

Generally, as H or G increase, the number of active APs, the link speed, the minimum average throughput, and the overall throughput are increased. Thus, one way of using the proposed algorithm is to keep its applications while increasing either H or G until the number of APs exceeds the limit that has been given by considering the costs, and to use the solution that satisfies the constraints with the largest H and G.

Table 4: Simulation results after host association finalization in *random instance*.

Link speed threshold H (Mbps)		Average throughput threshold G (Mbps)		
		0	10	20
20	Avg. of E_1	4.00	6.00	11.00
	Avg. of $1/E_2$	7.14	10.82	20.80
	Avg. of min. avg. throughput (Mbps)	6.97	10.63	20.55
	Avg. of overall throughput (Mbps)	223.08	340.23	657.51
40	Avg. of E_1	4.00	6.00	11.00
	Avg. of $1/E_2$	7.33	10.88	20.97
	Avg. of min. avg. throughput (Mbps)	7.18	10.70	20.72
	Avg. of overall throughput (Mbps)	229.86	342.28	663.10
60	Avg. of E_1	9.00	9.00	11.00
	Avg. of $1/E_2$	11.55	11.55	21.19
	Avg. of min. avg. throughput (Mbps)	11.38	11.38	20.99
	Avg. of overall throughput (Mbps)	364.04	364.04	671.76

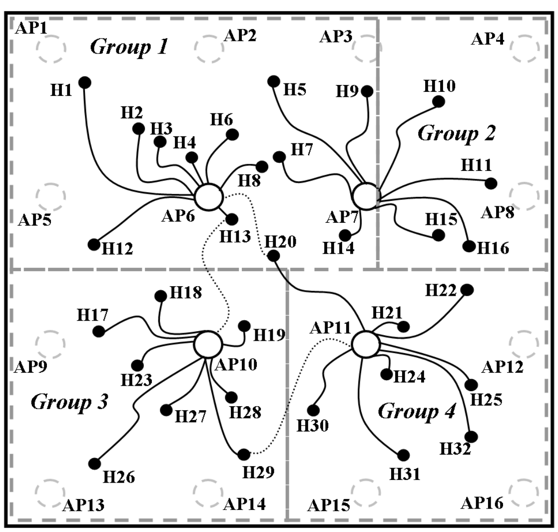

Figure 6: Four Active APs and host associations for $G = 0$ and $H = 20$ in *random instance*.

5.2.5 Evaluation of Algorithm Overall Performance

To evaluate the overall performance of the algorithm, we discuss the difference between the results in case of the topology in Figure 5 and the results in Table 4. The topology in Figure 5 has 16

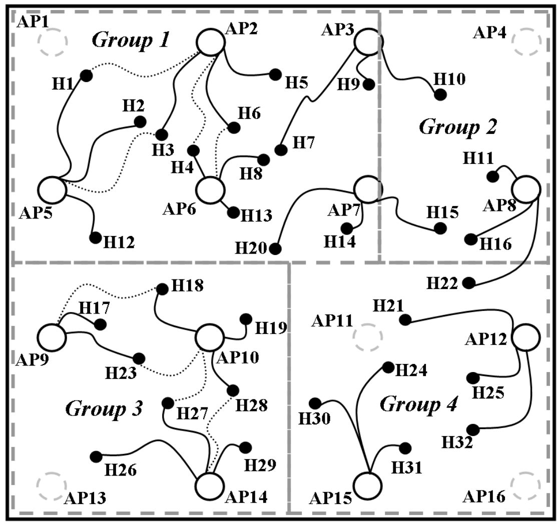

Figure 7: 11 Active APs and host associations for $G = 20$ and $H = 20$ in *random instance*.

APs ($E_1 = 16$). We computed the value of $1/E_2$ as 8.51, and generated the minimum average throughput and the overall throughput using the WIMNET simulator as $8.36Mbps$ and $267.43Mbps$ respectively. By comparing these results for the case with $G = 0$ with $H = 20$ in Table 4, our algorithm can reduce the number of active APs (E_1) by 75% (from 16 to 4), whereas the average overall throughput decreases only by 16.6% (from $267.43Mbps$ to $223.08Mbps$). Besides, for the cases with $G = 10$ with $H = 20$ or 40, our algorithm can reduce the number of active APs by 62.5% (from 16 to 6), whereas the average overall throughput increases by more than 25% (from $267.43Mbps$ to $340.23Mbps$ or $342.28Mbps$). Moreover, for the case with $G = 20$ with $H = 60$ in Table 4, our algorithm can reduce the number of active APs by 31.25% (from 16 to 11), whereas the average overall throughput increases by 152% (from $267.43Mbps$ to $671.76Mbps$).

The topology in Figure 6 illustrates that every AP is associated with eight hosts, while the topology in Figure 5 shows that **AP6** is associated with eight hosts while most APs are associated with only one host. Our algorithm improves the throughput performance by balancing loads among the active APs while reducing the number of active APs.

5.3 Simulation Results for Cafeteria Instance

Then, we consider another instance modeling a large-size *cafeteria* as a more common WLAN environment.

5.3.1 Simulation Instance

Figure 8 illustrates the topology in this *cafeteria instance*. Here, 3×3 (= 9) APs and 8×8 (= 64) seats are regularly placed with the same interval in the $35m \times 35m$ field. A *seat* represents a place for a user to stay and use a personal computer or a smart phone for the Internet access using WLANs. Only one group exists in this cafeteria.

To evaluate the extended algorithm, we generated 30 cases of host locations by randomly selecting seats with the 50% probability, assuming that usually, the seats are not fully occupied and selected by users randomly. Also, we assume that every time users move, the active APs are stopped and the new APs are turned on. However, in real situations, some APs cannot be turned off if they are currently associated with active hosts in the service area. Thus, the optimization of active APs under this constraint should be considered, which will be in our future studies.

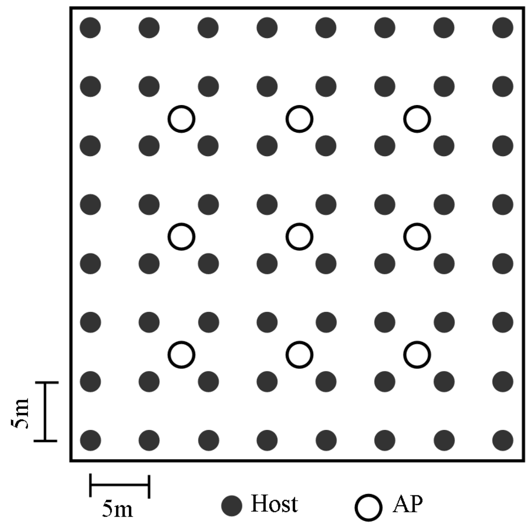

Figure 8: Cafeteria instance.

Table 5: Simulation results for 30 different host locations in *cafeteria instance*.

Link speed threshold H (Mbps)		Average throughput threshold G (Mbps)		
		0	10	20
50	Avg. of E_1	1.00	4.00	8.00
	Avg. of $1/E_2$	2.30	10.50	21.43
	Avg. of min. avg. throughput (Mbps)	2.19	10.27	21.12
	Avg. of overall throughput (Mbps)	69.96	328.61	675.79
70	Avg. of E_1	3.97	4.03	8.00
	Avg. of $1/E_2$	10.30	10.51	21.57
	Avg. of min. avg. throughput (Mbps)	10.07	10.27	21.25
	Avg. of overall throughput (Mbps)	322.14	328.69	679.96

5.3.2 Simulation Results

Table 5 shows the average values of E_1 and $1/E_2$ of the solutions found by the algorithm, and the average values of minimum average throughputs and overall throughputs obtained by simulations using the WIMNET simulator among the 30 cases, when we set $G = 0, 10, 20$ and $H = 50, 70$. These results confirm that our algorithm can satisfy the *average throughput constraint* for any case, and $1/E_2$ is the good estimation to the minimum average throughput even for different host locations.

In Table 5, with $H = 50$ and no constraint on average throughput per host ($G = 0$), the algorithm results in only one active AP that is associated with 32 hosts and provides a very small average throughput per host even if the high link speed threshold $50Mbps$ is considered. The link

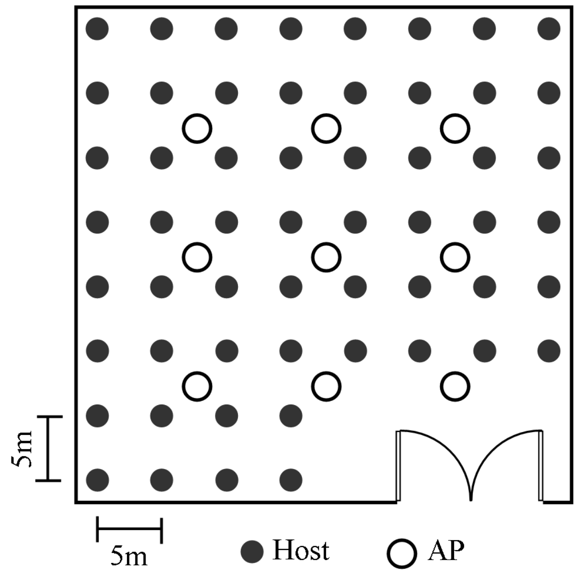

Figure 9: Biased cafeteria instance.

speed constraint ensures that any link between a host and its associated AP satisfies the minimum link speed when only one host is connected to the AP. However, it does not ensure the minimum throughput of a link when all the associated hosts are communicating at the same time. To solve this problem, the average throughput constraint is newly considered to ensure that any AP satisfies the given minimum average throughput per host when all the hosts are communicating. Then, the extended algorithm can successfully find the active APs and host associations that satisfy this constraint as in Table 5 for $H = 50$ with $G = 10$ or 20.

5.4 Simulation Results for Host Mobility

Hosts in a WLAN are usually not static. They may frequently join or leave the network. However, the application of the AP aggregation algorithm and the change of active APs and host associations for any host change is not realistic, because it may take time and suspend communications of hosts. In this paper, to deal with the host mobility, we consider that the active APs and the host-AP associations are fixed to the initial solution that is obtained by applying the algorithm to the topology where every host exists, and any joining host is associated with the AP specified in the initial solution during operations.

5.4.1 Simulation Instance

To evaluate the effectiveness of our approach for the host mobility, we consider the *biased cafeteria instance* in Figure 9 that contains 9 APs and 56 hosts. To avoid simple optimal solutions due to the symmetry in *cafeteria instance*, eight seats near the entrance are removed. For the host mobility, we generated 30 different cases of host locations by randomly selecting seats with the 50% probability.

5.4.2 Simulation Results

Then, we compare the performance of four cases: 1) the *proposal* case where the active APs and the host associations are fixed to the initial solution found by the algorithm, 2) the *optimal* case where at any change of hosts, the algorithm is applied and the topology is reconstructed, 3) the *manual* case where four active APs are selected symmetrically and any host is associated with the nearest active AP, and 4) the *reference* case where the four active APs in case 1 are selected and any host is associated with the nearest active AP. Table 6 shows the average of the CPU time, E_1 and $1/E_2$,

Table 6: Simulation results for 30 different host locations in *biased cafeteria instance*.

	Case 1 (proposal)	Case 2 (optimal)	Case 3 (manual)	Case 4 (reference)
Avg. of CPU time (sec)	(14.741)	7.011	-	-
Avg. of E_1	4.00	3.83	4.00	4.00
Avg. of $1/E_2$	10.48	11.41	9.27	9.96
Avg. of min. avg. throughput (Mbps)	10.23	11.15	9.03	9.72
Avg. of overall throughput (Mbps)	281.61	310.05	247.29	266.44

and the minimum average throughputs, and the overall throughputs for the 30 cases. Here, we set $G = 10$ and $H = 50$, and assign a different channel to any active AP.

For the proposal case, we applied the algorithm with $G = 5$ to generate the initial solution so that it can provide $10Mbps$ for the minimum average throughput per host when hosts exist with the 50% probability. For case 1 and case 4, the three corner APs except the one near the entrance, and the center AP at the third column are selected as active APs. For case 3, the four corner APs are selected as active APs. For case 2, active APs are selected depending on the hosts.

Table 6 shows that case 1 (proposal) provides better throughputs than case 3 (manual) and case 4 (reference), and case 4 does better throughputs than case 3. The difference between case 1 and case 2 (optimal) is smaller than that between case 1 and case 3. These facts justify that the proposed algorithm improves active AP selections and host associations than their simple selections. The CPU time for case 1 is $14.741sec$, which is necessary only once. The average CPU time for case 2 is $7.011sec$, which is necessary every time a host moves. These results indicate that even this simple approach using the AP aggregation algorithm can improve the performance. However, if the association of a joining host is optimized by considering AP loads, the performance can be further improved. The optimization of the joining host association will be in our future studies.

5.5 Simulation Results for Dormitory Instance

As a more practical and complex WLAN environment to evaluate the AP aggregation algorithm, we consider the *dormitory instance* that mimics one floor in a student dormitory. It contains multiple rooms separated by walls that have the different size and number of hosts.

5.5.1 Simulation Instance

Figure 10 illustrates the topology in the *dormitory instance*. This topology contains four six-person rooms, seven four-person rooms, one common toilet, and one common dining room. One AP is allocated in each student room, and two APs are in the dining room assuming six students are actively connecting with the Internet there. Thus, a total of 13 APs and 58 hosts in the same group may exist in this instance.

Each room is separated by concrete walls from the others. Due to measurement results in [21], the IEEE802.11n link speed is dropped by about 15% when the link passes one concrete wall. Because the WIMNET simulator that we use for simulations does not consider this wall effect, we replace one wall by the $10m$ free space. The maximum end-to-end distance of this dormitory instance is about $40m$. In this range, Figure 2 shows that the 15% link speed drop is roughly equivalent to the $10m$ increase of the link distance.

5.5.2 Simulation Results

For the dormitory instance, we use $G = 10$ for the *average throughput threshold* and $H = 20$ for the *link speed threshold* as typical ones. We assign a different channel to any active AP. Then, the AP aggregation algorithm found a solution of seven active APs of **AP1**, **AP4**, **AP7**, **AP8**, **AP9**,

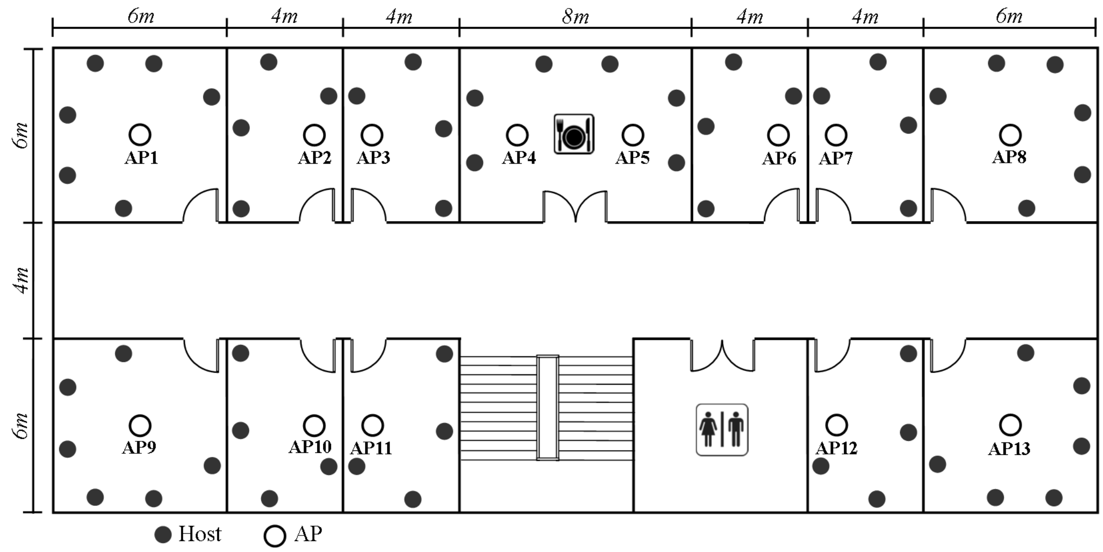

Figure 10: Dormitory instance.

AP10, and **AP13**. Here, we note that the estimated value of the minimum average throughput by $1/E_2$ is 10.14 whereas the minimum average throughput by the simulation is $9.87 Mbps$.

5.5.3 Evaluation of AP Selection and Host Association in Algorithm

To evaluate the effectiveness of the AP aggregation algorithm in terms of the active AP selection and the host association, we conducted the performance comparisons between the algorithm case and the manual case. For the manual case, we prepare the manual active AP selection and the manual host association. For the manual active AP selection, we selected the same number of active APs (seven APs) alternately from the lower side and the upper side of the field, so that any host can find an active AP with a short distance. Here, actually we selected **AP2**, **AP4**, **AP6**, **AP8**, **AP9**, **AP11**, and **AP12**. For the manual host association, we selected the nearest active AP for any host.

To verify the effectiveness of each selection procedure in our algorithm, we actually compared the performance with three cases. In case 1, both the active AP selection and the host association are manual. In case 2, the active AP selection is by the algorithm and the host association is manual. In case 3, the active AP selection is manual and the host association is by the algorithm. Table 7 shows $1/E_2$, the minimum average throughput, and the overall throughput for them. It indicates that the proposal case provides the better throughputs than any reference case. Thus, the effectiveness of both selection procedures in our algorithm is verified in this practical and complex instance.

5.5.4 Average Throughput Constraint Satisfaction by Manual Selection

Then, we examine the required number of active APs to satisfy the average throughput constraint with $G = 10$ by the manual selection. Table 8 shows the minimum average throughput and the overall throughput by the simulation when we manually generate a solution with 7, 8 and 9 active APs as the same way. It shows that nine active APs are necessary to satisfy $G = 10$, although it provides the better performance than the algorithm solution with seven active APs.

5.6 Discussion on Link Speed Threshold

One interesting feature of the results in Tables 4 and 5 is that even for the same number of active APs, the performance index (minimum average and overall throughputs) can be different depending

Table 7: Simulation results with $G = 10$ and $H = 20$ in *dormitory instance*.

		Case information				Results		
		AP selection	Host association	# of active APs (E_1)	Selected active APs	$1/E_2$	Min. avg. throughput (Mbps)	Overall throughput (Mbps)
Proposal		algorithm	algorithm	7	1, 4, 7, 8, 9, 10, 13	10.14	9.87	572.43
Reference cases	Case 1	manual	manual	7	2, 4, 6, 8, 9, 11, 12	7.29	7.09	411.46
Reference cases	Case 2	algorithm	manual	7	1, 4, 7, 8, 9, 10, 13	8.77	8.52	494.34
Reference cases	Case 3	manual	algorithm	7	2, 4, 6, 8, 9, 11, 12	9.99	9.72	563.87

Table 8: Simulation results for manual selections with increasing number of active APs in *dormitory instance*.

# of active APs	7	8	9
Selected active APs	2, 4, 6, 8, 9, 11, 12	1, 3, 5, 7, 8, 9, 11, 13	1, 3, 5, 7, 8, 9, 11, 12, 13
Min. avg. throughput (Mbps)	7.09	8.61	11.29
Overall throughput (Mbps)	411.46	499.38	654.68

on the *link speed threshold* H. For example, in the *random instance*, the performance index is better for $H = 40$ than $H = 20$, although the same number of active APs is used. Thus, H should be optimized when the number of active APs is fixed. This further extension of the AP aggregation algorithm will be also in our future studies.

6 Conclusion and Future Works

In this paper, we presented two extensions of the AP aggregation algorithm for large-scale wireless networks: 1) ensuring the *minimum average throughput* for any host by adding active APs, and 2) further optimizing host associations by changing multiple hosts simultaneously with the *host association finalization* phase. We verified the effectiveness through simulations in random, cafeteria, and dormitory instances using the WIMNET simulator. In future studies, we will consider the optimization of the *link speed threshold*, the minimum average throughput estimation under limited available channels, the algorithm extension under the existence of unstoppable active APs, the joining host association for host mobility, and evaluate the proposal in a variety of network instances and in real WLANs.

References

[1] Matthew Gast. *802.11 Wireless Networks: The Definitive Guide*. O'Reilly Media, Inc., 2nd edition, April 2005.

[2] Andrew S Tanenbaum. *Computer Networks*. Prentice Hall, 5th edition, October 2010.

[3] Steven Lanzisera, Bruce Nordman, and Richard E. Brown. Data network equipment energy use and savings potential in buildings. *Energy Efficiency*, 5(2):149–162, May 2012.

[4] Bernard Aboba. Virtual access points. *IEEE Contribution*, 802.11-03-154r1, May 2003.

[5] Takahiro Nagai and Hiroshi Shigeno. A framework of ap aggregation using virtualization for high density WLANs. In *Proceedings of Third International Conference on Intelligent Networking and Collaborative Systems (INCoS 2011)*, pages 350–355. IEEE, November–December 2011.

[6] Hiroko Shindo, Takahiro Nagai, and Hiroshi Shigeno. Virtual access point allocation method for high density WLANs. In *Proceedings of Sixth International Conference on Mobile Computing and Ubiquitous Networking (ICMU 2012)*, pages 69–76, May 2012.

[7] R. Mahindra, G.D. Bhanage, G. Hadjichristofi, I. Seskar, D. Raychaudhuri, and Y.Y. Zhang. Space versus time separation for wireless virtualization on an indoor grid. In *Proceedings of Next Generation Internet Networks (NGI 2008)*, pages 215–222, April 2008.

[8] Md. Ezharul Islam, Nobuo Funabiki, Toru Nakanishi, and Kan Watanabe. An access-point aggregation approach for energy-saving wireless local area networks. *IEICE Transactions on Communications*, E96-B(12):2986–2997, December 2013.

[9] Nobuo Funabiki, editor. *Wireless mesh networks*. InTech-Open Access Pub., January 2011. available at `http://www.intechopen.com/books/show/title/wireless-mesh-networks`.

[10] Amit P. Jardosh, Konstantina Papagiannaki, Elizabeth M. Belding, Kevin C. Almeroth, Gianluca Iannaccone, and Bapi Vinnakota. Green WLANs: On-demand WLAN infrastructures. *Mobile Networks and Applications*, 14(6):798–814, December 2009.

[11] Fatemeh Ganji, Lukasz Budzisz, and Adam Wolisz. Assessment of the power saving potential in dense enterprise wlans. In *Proceedings of 24th International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC 2013)*, pages 2835–2840. IEEE, September 2013.

[12] Xiang Ling and Kwan Lawrence Yeung. Joint access point placement and channel assignment for 802.11 wireless LANs. *IEEE Transactions on Wireless Communications*, 5(10):2705–2711, October 2006.

[13] Andreas Eisenblatter, Hans-Florian Geerdes, and Iana Siomina. Integrated access point placement and channel assignment for wireless LANs in an indoor office environment. In *Proceedings of International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2007)*, pages 1–10. IEEE, June 2007.

[14] Nobuo Funabiki, Junki Shimizu, Toru Nakanishi, and Kan Watanabe. A proposal of an active access-point selection algorithm in wireless mesh networks. In *Proceedings of 14th International Conference on Network-Based Information Systems (NBiS 2011)*, pages 112–117. IEEE, September 2011.

[15] Fengyuan Xu, Xiaojun Zhu, Chiu C Tan, Qun Li, Guanhua Yan, and Jie Wu. SmartAssoc: Decentralized access point selection algorithm to improve throughput. *IEEE Transactions on Parallel and Distributed Systems*, 24(12):2482–2491, December 2013.

[16] Sumiko Miyata, Tutomu Murase, and Katsunori Yamaoka. Novel access-point selection for user QoS and system optimization based on user cooperative moving. *IEICE Transactions on Communications*, 95(6):1953–1964, June 2012.

[17] Matthew Gast. *802.11n: A Survival Guide*. O'Reilly Media, Inc., March 2012.

[18] Working Group of the 802 Committee et al. IEEE standard for information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 5: Enhancements for higher throughput. *IEEE Std*, 802.11n:c1–502, October 2009.

[19] Nobuo Funabiki, Wataru Maruyama, Toru Nakanishi, and Kan Watanabe. An extension of routing tree algorithm considering link speed change in IEEE 802.11n protocol for wireless mesh network. In *Proceedings of 9th International Conference on Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC 2012)*, pages 600–605. IEEE, September 2012.

[20] David P Williamson and David B Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, April 2011.

[21] Tatsuya Takebayashi. Throughput measurements of IEEE802.11n wireless LAN and application to network design. Master's thesis, Graduate School of Natural Science and Technology, Okayama University, Japan, March 2012.