(302) Network

A Proposal of Preprocessing Stage for Active Access-Point Configuration Algorithm in Elastic WLAN System

Mousumi Saha†, Nobuo Funabiki††, Sumon Kumar Debnath†, and Md. Selim Al Mamun†
†{sahamousumi,debnathsumon ,mamun0013 }@s.okayama-u.ac.jp ††funabiki@okayama-u.ac.jp

ABSTRACT

We have studied the *active AP configuration algorithm* for the *elastic WLAN system* to optimize the network configuration by dynamically controlling the active *access points (APs)* and the host associations depending on network situations. However, this algorithm may take long CPU time and may not provide an optimal solution when the number of APs is large. In this paper, we propose the *preprocessing stage* for the algorithm. By selecting promising candidates for active APs, it can drastically reduce the search space. The simulation results in four network topologies show that the proposal improves the overall throughput by 1.56% and reduces the total CPU time by 28.13% on average.

Keywords

access point, active AP configuration algorithm, elastic WLAN system, preprocessing.

1 INTRODUCTION

Nowadays, *wireless local-area networks (WLANs)* have gained popularity in many places including schools, offices, and public spaces [1]. In WLAN, hosts can access the Internet through associations with the *access points (APs)* using wireless links. The distribution of user hosts in WLAN is usually non-uniform, and their traffic demands fluctuate with time [1] [2]. Besides, APs are often placed at random locations, which may cause poor performances due to signal interferences among them.

To solve the above mentioned problems, we have proposed the *active AP configuration algorithm* for the *elastic WLAN system* that dynamically optimizes the network configuration by activating or deactivating allocated APs and changing host associations according to the network conditions [3].

However, the AP configuration algorithm may take long CPU time and may not provide an optimal solution when the number of APs is large, due to the nature of the heuristic algorithm. It is necessary to reduce the number of APs to be given to the algorithm.

In this paper, we propose the *preprocessing stage* for the active AP configuration algorithm to achieve it by selecting only promising candidates. In this stage, first, the minimum number of active APs L required to satisfy the necessary throughput is estimated. Then, the number of candidate APs K is estimated from L. Finally, K APs are selected as promising candidates for active APs, which are used as the input to the active AP configuration algorithm. The effectiveness of this proposal is verified through simulations in three network topologies. It is found that the proposal improves the overall throughput by 1.56% and reduces the total CPU time to run the whole algorithm by 28.13% on average.

The rest of this paper is organized as follows: *Section* **2** reviews our previous works related to this paper. *Section* **3** proposes the preprocessing stage for the active AP configuration algorithm. *Section* **4** evaluates the proposal through simulations. Finally, *Section* **5** concludes this paper with future works.

2 REVIEW OF PREVIOUS WORKS

In this section, we review our previous works related to this paper.

2.1 Overview of Elastic WLAN System

The *elastic WLAN system* dynamically controls the active APs and the hosts association according to the output of the *active AP configuration algorithm*. The implementation of the elastic WLAN system testbed adopts a server to manage the necessary information and control the APs and the hosts. This server has the administrative access rights to all the hosts and APs in the network.

2.2 Overview of Active AP Configuration Algorithm

The active AP configuration algorithm consists of eight steps [3].

1. **Preparation**: The link speed for any possible pair of an AP and a host is estimated using the throughput estimation model [7].

2. **Initial Solution Generation**: An initial solution is derived using a greedy method [5] of selecting the active APs and host associations that can cover the largest number of hosts. Then, the number of active APs is represented by E_1.

3. **Host Association Improvement**: The average host throughput E_2 is calculated for the initial solution by:

$$E_2 = \min_j [TH_j] \quad (1)$$

where TH_j represents the average host throughput for the jth AP and is defined by:

$$TH_j = \frac{1}{\sum_k \frac{1}{s_{jk}}} \tag{2}$$

where s_{jk} represents the link speed between the jth AP and the kth host. Then, this solution is improved by finding better host associations.

4. **AP Selection Optimization**: E_1 and E_2 are jointly optimized by applying a local search method [6] of randomly changing active APs and finding host associations to minimize them.

5. **Link Speed Normalization**: The fairness criterion is applied here in adjusting the link speed, when the total expected bandwidth B^e exceeds the given total bandwidth B^a:

   (a) Calculate B^e by taking the summation of the throughputs of all the APs.

   (b) If $B^e > B^a$, adjust each AP-host link speed as:

   $$\hat{s}_{ij} = s_{ij} \times \frac{B^a}{B^e} \tag{3}$$

   where $\hat{s}_{ij}$ is the adjusted link speed, which will be used in the following steps.

6. **Constraint Check**: When the *minimum host throughput constraint* is satisfied or all the APs in the network have been activated, go to the next step. Otherwise, go to step 3.

7. **Channel Assignment** One channel is assigned to each active AP such that the total interfered communication time E_3 is minimized:

   $$E_3 = \sum_{i=1}^{M} [IT_i] = \sum_{i=1}^{M} \left[\sum_{\substack{k \in I_i \\ c_k = c_i}} T_k \right] \tag{4}$$

   where M represents the number of active APs, IT_i does the interfered communication time, T_i does the total communication time, I_i does the set of interfered APs, and c_i does the assigned channel for AP_i respectively.

8. **Channel Load Averaging** The loads of the APs are averaged among the different channels.

3 PROPOSAL OF PREPROCESSING STAGE

In this section, we propose the preprocessing stage for the active AP configuration algorithm.

3.1 Estimation of Candidate AP Number

In the preprocessing stage, an estimated number of promising candidates for active APs are selected for the input to the active AP configuration algorithm. To estimate this number K properly, the minimum number of active APs L that are necessary to satisfy the host throughput constraint is estimated first. Then, L can be estimated from the minimum host throughput threshold G, the number of hosts N, and the maximum possible link speed TH_{max} in the network field by:

$$L < (G \times N)/TH_{max}. \tag{5}$$

where TH_{max} is generally $100Mbps$ for an *IEEE 802.11n* link, which has been observed in our experiments. Therefore, L is estimated by:

$$L = (G \times N)/100. \tag{6}$$

Then, K is estimated from L by:

$$K = \beta \times L \tag{7}$$

where β represents a given constant parameter. Through simulations, it is found that the proper value of β is different depending on the minimum host throughput threshold G: $\beta = 4.3$ for $G \leq 10$, and $\beta = 3.1$ otherwise.

3.2 Procedure of Preprocessing Stage

The procedure of the preprocessing stage for the active AP configuration algorithm is presented based on the exhaustive search in this paper.

1. **Calculation of Link Speed:** The link speed (throughput) for each link between each of the given M candidate APs and each of the given N hosts is estimated using the throughput estimation model.

2. **Generation of Possible AP Combinations:** The total of ${}_MC_K$ possible combinations of K APs are generated, by selecting K APs from the M candidates.

3. **Initialization of Objective Function:** The best-found objective functions E^{best} and E_2^{best} are initialized by 0, where E^{best} represents the best-found value of the summation of the bottleneck host throughput and E_2^{best} does the best-found value of the average host throughput for K APs.

4. **Examination of New AP Combination:** For each combination of K APs, the following procedure is applied:

   (a) The candidate AP that has the largest link speed is selected for the associated AP of each host.

(b) The best-found objective functions E^{best} and E_2^{best} are updated and the current AP selection is saved in memory, if at least one of the two best found objective function, E^{best} or E_2^{best} is improved and another one remains the same by the current AP selection and the host associations.

5. **Termination check:**

   If all the combinations of K APs is examined, the procedure is terminated, and the selected K APs is used for the input to the active AP configuration algorithm. Otherwise, go to step 4 to examine another new combination.

4 EVALUATIONS

In this section, we evaluate the proposed preprocessing stage through simulations in three network topologies using hardware and software in Table 1. To verify the effectiveness, the minimum average host throughput E_2, the overall throughput of all the hosts, and the total CPU time to run the whole algorithm are compared between two cases: with and without the processing stage for the active AP configuration algorithm. To investigate the change of the performance under different throughput requests, (G) is changed from 5 to 20.

Table 1: Simulation environment.

Simulator	WIMNET Simulator [8]
interface	IEEE 802.11n
CPU	Intel Core i7
memory	4GB
OS	Ubuntu LTS 14.04

4.1 Evaluation in Topology I

As the first topology, Figure 1 illustrates *Topology I* that is composed of 25 hosts, 20 candidate APs, and two $50m \times 50m$ rooms. The circles and squares represent the AP and host locations respectively.

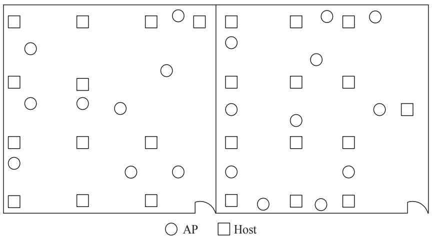

Figure 1: Topology I with 25 hosts and 20 candidate APs.

Table 2 shows the simulation results for *Topology I*. The proposal improves the overall throughput by 2.22% and decreases the CPU time by 62.46% on average.

Table 2: Simulation results for *Topology I*.

G (Mbps)	# of active APs	without proposal: min. host through. (Mbps)	without proposal: overall through. (Mbps)	without proposal: CPU time (s)	with proposal: min. host through. (Mbps)	with proposal: overall through. (Mbps)	with proposal: Exhaustive CPU time (s)	with proposal: APC CPU time (s)	with proposal: Total CPU time (s)
5	3	5.22	130.16	20.07	5.38	133.66	0.347	1.43	1.78
10	6	12.10	300.52	53.87	12.59	312.86	0.266	17.55	17.82
15	7	15.02	373.58	100.9	15.04	373.98	0.266	30.57	30.84
20	9	20.87	519.51	162.69	21.41	532.67	0.055	76.23	76.29

4.2 Evaluation in Topology II

As the second topology, Figure 2 illustrates *Topology II* that is composed of 40 hosts, 25 candidate APs, three $7m \times 6m$ rooms, and one $3.5m \times 6m$ room. This topology models the third floor of Engineering Building#2 at Okayama University.

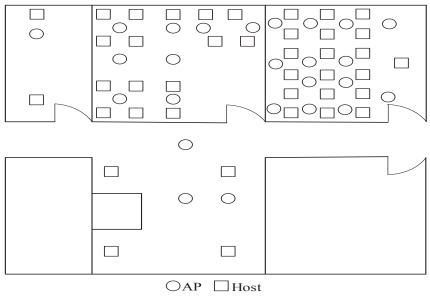

Figure 2: Topology II with 40 hosts and 25 candidate APs.

Table 3 shows the simulation results for *Topology II*. The proposal improves the overall throughput by 0.66% and decreases the CPU time by 31.21% on average.

Table 3: Simulation results for *Topology II*.

G (Mbps)	# of active APs	without proposal: min. host through. (Mbps)	without proposal: overall through. (Mbps)	without proposal: CPU time (s)	with proposal: min. host through. (Mbps)	with proposal: overall through. (Mbps)	with proposal: Exhaustive CPU time (s)	with proposal: APC CPU time (s)	with proposal: Total CPU time (s)
5	3	5.64	224.32	65.51	5.73	229.94	4.242	2.41	6.65
10	6	11.64	472.27	256.32	11.84	482.41	1.555	85.58	87.14
15	9	17.40	724.39	439.95	17.51	723.87	0.665	217.64	218.31
20	11	21.70	877.38	679.32	21.70	877.38	0.011	679.32	679.33

4.3 Evaluation in Topology III

As the third topology, Figure 3 illustrates *Topology III* that is composed of 40 hosts, 25 candidate APs, two $60m \times 45m$ rooms, and six $30m \times 45m$ rooms.

Table 4 shows the simulation results for *Topology III*. The proposal improves the overall throughput by 0.50% and decreases the CPU time by 41.38% on average.

4.4 Evaluation in Topology IV

As the forth and largest topology, Figure 4 illustrates *Topology IV* that increases the number of candidate APs to 33.

Table 5 shows the simulation results for *Topology IV*. The proposal improves the overall throughput by 2.86%, but increases the CPU time by 6.9% due to the large number of combinations of selecting APs.

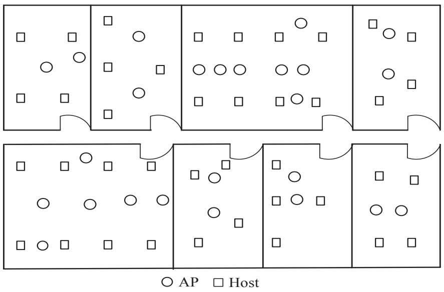

Figure 3: Topology III with 40 hosts and 25 candidate APs.

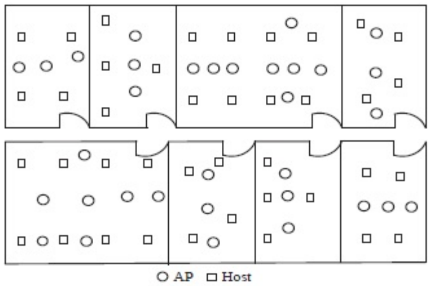

Figure 4: Topology IV with 40 hosts and 33 candidate APs.

Table 4: Simulation results for *Topology III*.

G (Mbps)	# of active APs	without proposal			with proposal				
		min. host through. (Mbps)	overall through. (Mbps)	CPU time (s)	min. host through. (Mbps)	overall through. (Mbps)	Exhaustive CPU time (s)	APC CPU time (s)	Total CPU time (s)
5	7	6.31	250.74	348.85	6.55	260.54	4.242	4.58	8.82
10	10	13.12	522.53	690.52	13.15	523.76	1.555	220.88	222.43
15	14	15.64	621.97	1320.83	15.64	621.86	0.665	441.45	442.11
20	17	20.07	800.32	1700.58	20.07	800.32	0.011	1700.57	1700.59

Table 5: Simulation results for *Topology IV*.

G (Mbps)	# of active APs	without proposal			with proposal				
		min. host through. (Mbps)	overall through. (Mbps)	CPU time (s)	min. host through. (Mbps)	overall through. (Mbps)	Exhaustive CPU time (s)	APC CPU time (s)	Total CPU time (s)
5	4	6.36	252.72	375.67	6.72	267.05	55.61	6.58	62.19
10	7	10.64	423.98	801.21	11.32	453.95	2077.49	267.05	2211.77
15	9	15.71	626.28	1701.78	15.91	634.54	1596.69	342.63	1939.33
20	10	20.18	815.73	2074.27	20.04	823.79	33.35	1047.05	1081.13

5 CONCLUSION

This paper proposed the preprocessing stage for the AP configuration algorithm in the elastic WLAN system. By selecting promising candidates for active APs, it can reduce the search space. The simulation results in four network topologies showed that the proposal improved the overall throughput by 1.56% and reduced the total CPU time by 28.13% on average. However, the CPU time increased in the largest instance due to the rapid increase of possible combinations, which should be reduced in our future works.

Acknowledgments

This work is partially supported by JSPS KAKENHI (16K00127).

References

[1] M. Balazinska and P. Castro, "Characterizing mobility and network usage in a corporate wireless local-area network," in Proc. Int. Conf. Mob. Syst. Appl. Ser., pp. 303-316, 2003.

[2] K. Mittal, E. M. Belding, and S. Suri, "A game-theoretic analysis of wireless access point selection by mobile users," Comput. Commun., vol. 31, no. 10, pp. 2049-2062, Jan. 2008.

[3] M. S. A. Mamun, N. Funabiki, K. S. Lwin, M. E. Islam, and W.-C. Kao, "A channel assignment extension of active access-point configuration algorithm for elastic WLAN system and its implementation using Raspberry Pi," Int. J. Netw. Comput., vol. 7, no. 2, pp. 248-270, July, 2017.

[4] K. Farkas, A. Huszak, and G. Godor, "Optimization of Wi-Fi access point placement for indoor localization," Network. Commun., vol. 1, no. 1, pp. 28-33, July 2013.

[5] L. A. Wolsey, "An analysis of the greedy algorithm for the submodular set covering problem," Combinatorica, vol.2, no.4, pp. 385-393, Dec. 1982.

[6] D. P. Williamson and D. B. Shmoys, "The design of approximation algorithms," Cambridge Univ. Press, Apr. 2011.

[7] K. S. Lwin, N. Funabiki, C. Taniguchi, K. K. Zaw, M. S. A. Mamun, M. Kuribayashi, and W.-C. Kao, " A minimax approach for access point setup optimization in IEEE 802.11n wireless networks, " Int. J. Netw. Comput., vol. 7, no. 2, pp. 187-207, July 2017.

[8] N. Funabiki ed., "Wireless mesh networks," InTech-Open Access Pub., Jan. 2011, `http://www.intechopen.com/books/wireless-mesh-networks`, Access July 29, 2019.