

# Self-Optimizing Scanning Parameters for Seamless Handover in IEEE 802.11 WLAN

Lina Hao, Bryan Ng, Ying Qu  
School of Engineering and Computer Science,  
Victoria University of Wellington, New Zealand  
{lina.hao, bryan.ng, ying.qu}@ecs.vuw.ac.nz

**Abstract**—Within IEEE 802.11 Wireless Local Area Networks (WLANs), a handover between wireless Access Points (APs) is inevitable for the mobile stations to maintain network connectivity as they move. Handover occurs frequently due to the limited coverage range of APs and the mobility of stations (STAs). High handover latency causes interruption to data flow and degrades the quality of service (QoS) experienced by the mobile stations. Therefore, minimizing handover latency is critical for maintaining seamless communications and improved QoS. In this paper, we build upon recent advances in mobile networks whereby Self-Organizing Network (SON) has been shown to work well in handling mobility. Inspired by SON, we propose an algorithm that combines: (i) scanning timers optimized with Genetic Algorithm (GA) and (ii) self-adapting timers to reduce the handover latency for 802.11 WLANs. We use the IEEE 802.11k standard to retrieve radio parameters from measurement reports and feed these parameters to a GA to find the best fit values to reduce scanning delay. We compare the proposed algorithm performance with IEEE 802.11 standard and our previous work neighbor list mechanism (NLM). Evaluation results demonstrate that our proposed algorithm reduce the handover latency by 78.79% compared with standard 802.11, the packet loss is 3.37% lower than standard 802.11 while the throughput improves up to 3.93%.

## I. INTRODUCTION

The IEEE 802.11 Wireless LANs are being used ubiquitously in campuses, enterprises and also as a predominant wireless access network. Its advantages over other wireless networks (e.g. 3G, LTE), such as low cost, easy deployment and high bandwidth, will make it an important component in next generation networks. New and emerging uses of 802.11 WLANs such as backhaul connectivity for Internet-of-Things, vehicular communications and smart energy grids have been investigated in the literature.

Some of the new found uses of 802.11 WLANs may not materialize in practice in part due to relatively poor support for mobility in 802.11 WLANs (e.g. compared to LTE or 3G networks). For example, in 802.11 WLANs, mobility related parameters such as channel scanning time and thresholds for switching over between wireless access points are manually adjusted to achieve high level of network performance (i.e. high throughput and low delay).

Techniques from LTE such as Self-Organizing Network (SON), where the network tunes parameters automatically based on ongoing measurements may help improve mobility management in 802.11 WLANs. A challenge is to adapt the salient features of LTE SON (and the best practices therein)

to improve 802.11 WLANs handover management for better QoS in 802.11 WLANs.

The handover is a process to transfer ongoing calls or data sessions when a mobile station (STA) moves from one wireless access point (AP) to another AP. However, there is no specific handover algorithm to support mobility management in the IEEE 802.11 standard. The IEEE 802.11 WLANs only support a STA to change its association from one access point to another owing to poor link quality. A basic 802.11 WLANs handover process includes three phases: scanning/probe, re-authentication and re-association. In the three phases, the scanning delay is the dominant component which accounts for more than 90% of the overall handover delay [1].

Unsurprisingly, many handover and mobility management solutions have been proposed in the literature to reduce scanning delay. Most of the solutions are based on the assumption that the probe timers *MinChannelTime* and *MaxChannelTime* are fixed. The timer *MinChannelTime* is the minimum waiting time a STA undergoes before considering that the channel is empty while the timer *MaxChannelTime* is the maximum waiting time after a Probe Response has been successfully received.

In this paper, we propose a SON inspired approach using self-optimizing scanning timers optimized by a Genetic Algorithm(GA). These self-optimizing adaptive timers are dynamically adjusted based on real-time network conditions to reduce MAC layer handover latency. Each STA is made aware of the scanning timers for each channel by a centralized controller. Therefore, the STA probes a set of channels and APs with adaptive timers instead of fixed values. We expect that the proposed algorithms to reduce the number of scanned channels, probed access points and unnecessary waiting time of probe responses of non-adjacent APs or APs with poor QoS.

To our best knowledge, this paper is the first to optimize all the scanning parameters together to access channels for each AP to improve handover performance in 802.11 wireless networks. The rest of the paper is organized as follows. Section II presents the background and related work. Section III describes our proposed algorithm. Section IV describes the results of our simulation. Finally, the conclusions are shown in Section V.

## II. BACKGROUND AND RELATED WORK

Previous studies have examined how to reduce scanning delay to provide seamless handover. For example, the work in S. Shin et al. [2] introduced a data structure called a channel mask, to reduce the number of channels scanned. They used the selective scanning algorithm to ensure that only a selected subset of channels will be scanned in future handovers. There are some concerns with the approach taken in [2] that the selective scan may miss certain channels because they are not scanned until a timeout.

The authors in [3] proposed using a BIGAP soft-handover. There is no need for a STA to scan all channels; instead a single channel switch reduces the scanning delay. However, the overall throughput degrades to some extent as the architecture is prone to collisions. Hence, a collision-free channel assignment is needed to increase the throughput of the network. In another study [4], the authors developed a fast handover scheme called Proactive Scan. It decouples the time-consuming channel scanning from the actual handover, and to eliminate channel scan delay by doing scan early. The scanning is performed pro-actively without human-detectable interruptions to ongoing traffic.

Most of the above solutions are based on the assumption that the probe timers *MinChannelTime* and *MaxChannelTime* are fixed. A few researchers have studied how to dynamically optimize the scanning parameters such as *MinChannelTime* or *MaxChannelTime* to reduce the probe waiting time. R. Pazzi et al. [5] and A. Boukerche et al. [6] proposed scanning algorithms adapted to the channel timers. The timers are dynamically adjusted based on the predicted channel information. However, the ideas in both [5] and [6] predict the target handover AP probability on each channel based on the assumption that the information collected from the last channel scan is accurate. This assumption is overly burdensome and requires significantly higher computational complexity because it maintains the state of the channel for each known AP.

The authors in [7] presented an extensive analysis of the scanning process by mean of simulation and real testbed. They observe that there is not only a single fixed optimal pair(*MinChannelTime* and *maxChannelTime*) that always gives the best scanning performance in all deployment, which is consistent with earlier studies [8] [2]. The algorithm in [7] use a simple adaptation method for adjusting the timer values in order to increase the discovery rate. The work in [7] departs from the trend of increasingly complex algorithms and argues for a simple yet practical solution.

In a recent paper [9], a novel approach based on multi-objective optimization approach to obtain the optimal number of channels to scan was developed. The optimal channel scanning sequence and its correspondent scanning timers including *MinChannelTime* and *MaxChannelTime* are used as optimization parameters. The results show that the adaptive scanning strategies better manage the performance trade-off and allow different application profiles to match with

specific scanning latency. However, the author did not consider other scanning parameters such as channel switching time and probe request interval and so on.

From the survey of the literature, few previous works have optimized the scanning parameters and the number of channels together to reduce both switching time and probe waiting time. In this paper, we optimize all the affected scanning parameters together to improve handover performance in 802.11. These parameters include *MinChannelTime*, *MaxChannelTime*, channel switching time, probe request interval, the number of channel scanned and channel sequence. We use a genetic algorithm that takes into account the loads of the APs and SNR to get a set of pre-configured scanning timers that reduce the handover time. These values are dynamically adjusted based on the real-time network feedback. Our proposed self-optimizing timers can reduce the probe delay significantly with the goal of reducing handover latency and facilitating seamless handover.

## III. PROPOSED SCHEME

In this section, we detail our approach to optimizing scanning timers. We articulate the problem of handover delay and present the optimization objective, the problem solutions and algorithms. The proposed SON approach uses a Genetic Algorithm (GA) to optimize scanning parameters and configure the scanning parameters for each STA before the scan process is initiated. Once the GA finds the optimal initial values of the scanning parameters, the table of GA optimized scanning parameters and channels are stored in the AP controller (APC). The GA parameters are refreshed as STAs scan new APs or APs go out of range. Hence, GA parameters are unlikely to be outdated. Moreover, the scanning timers can be further adjusted by the adaptive timers. The motivation behind using GA is to find the best candidate AP instead of scanning all available channels by a STA, the APC maintains an up-to-date list with GA optimized timers for each channel.

The network scenario used throughout this paper is shown in Fig. 1. It depicts that a centralized controller connected to all the APs. Once the STA detects that the current associated AP's signal strength is lower than the predefined threshold, the active scanning is initiated. The tables of GA optimized scanning parameters and channels are sent to the STA by the controller, and then the STA starts to search for the best candidate AP using the optimized parameters.

### A. Problem formulation

The total handover time includes scanning time, re-association time and re-authentication time [6]. Therefore, the handover latency for a handover in 802.11 WLANs can be calculated by Equation 1,

$$\sum_{i=1}^N T_{Handover(i)} = \sum_{i=1}^N \{T_{sc(i)} + T_{au(i)} + T_{as(i)}\}, \quad (1)$$

where  $T_{Handover(i)}$  is the handover latency of each channel  $i$ ;  $T_{sc(i)}$ ,  $T_{as(i)}$  and  $T_{au(i)}$  are scanning delay, association delay and authentication delay of each channel, respectively.

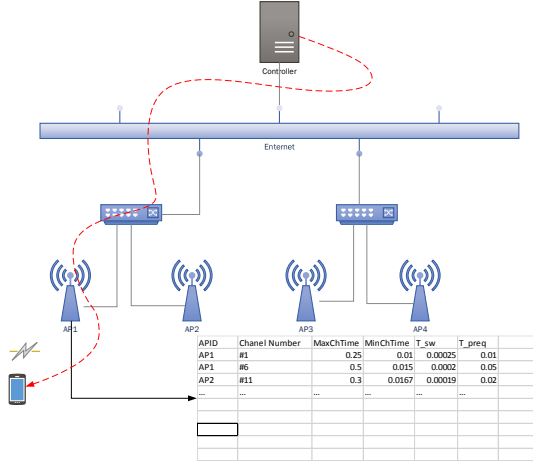


Fig. 1. Proposed SON network architecture

As the scanning delay is the primary concern accounting for 90% of the overall handover delay [1], our aim is to focus on reducing the scanning delay. In active scan, the scan time includes the probe time that is the time used to broadcast probe messages on the  $i$ -th channel, the switching time that is the time used by the STA to switch from current channel to the next channel and waiting time that is the time used for collecting the response messages [6]. The scan delay can be calculated using Equation 2,

$$\begin{aligned} \sum_{i=1}^N T_{sc}(i) &= \sum_{i=1}^N \{T_{Preq}(i) + T_{wt}(i) + T_{sw}(i)\} \\ &= \sum_{i=1}^N \{T_{Preq}(i) + p_e(i) * t_{min}(i) + (1 - p_e(i)) * t_{max}(i) \\ &\quad + T_{sw}(i)\}, \end{aligned} \quad (2)$$

where the number of channels is denoted by  $N$  and the probability of the  $i$ -th channel is empty (given by  $p_e(i)$ );  $T_{Preq}(i)$  denotes probe request time on channel  $i$ ;  $T_{sw}(i)$  denotes the switching time between channels;  $T_{wt}(i)$  denotes the probe response waiting time on channel  $i$ ;  $MinChannelTime$  of  $i$ -th channel is denoted as  $t_{min}(i)$  and  $MaxChannelTime$  of  $i$ -th channel is denoted as  $t_{max}(i)$ .

If the probe time is longer on each channel, the probability of discovered AP will be higher. Therefore, we assume the scanning process is exponential distribution, the probability density function(PDF) as below:

$$f(t) = \lambda e^{-\lambda t}, \quad (3)$$

where  $\lambda$  is the channel load,  $t$  is the probing time. The probability of AP discovery within  $t_{min}(i)$  of each channel can be calculated by Equation 4:

$$P_{min}(0 \leq t \leq t_{min}(i)) = \int_0^{t_{min}(i)} \lambda e^{-\lambda t} dt = 1 - e^{-\lambda t_{min}(i)}. \quad (4)$$

Therefore, the average discovery number of APs within  $t_{min}(i)$  is calculated by Equation 5:

$$N_{Min} = P_{min} * N_{ap}, \quad (5)$$

where  $N_{ap}$  is the discovery number of APs over each scanning time  $t_{max}(i)$  on channel  $i$ .

The average discovery rate of APs over a duration of  $t_{max}(i)$  is expressed as follows:

$$\begin{aligned} D &= \frac{N_{Min}}{t_{min}(i)} + \frac{N_{ap} - N_{Min}}{t_{max}(i) - t_{min}(i)} \\ &= \frac{P_{min} * N_{ap}}{t_{min}(i)} + \frac{(1 - P_{min}) * N_{ap}}{t_{max}(i) - t_{min}(i)}. \end{aligned} \quad (6)$$

Let  $A_i(ap)$  denote a set of available APs on channel  $i$  and  $S_i(snr)$  denote the Signal-to-noise (SNR) value of each AP whereby AP is indexed by  $i$  such that  $1 \leq i \leq n$  and  $i \in \mathbb{Z}^+$ . The set of AP and SNR are expressed as follows:

$$A_i(ap) = \{ap_{i,1}, \dots, ap_{i,k}, \dots, ap_{i,n}\}, ap_{i,k} \in N_i, \quad (7)$$

$$S_i(snr) = \{snr_{i,1}, \dots, snr_{i,k}, \dots, snr_{i,n}\}, snr_{i,k} \in N_i, \quad (8)$$

where  $ap_{i,k}$  denotes the AP of channel  $i$  with index  $k$ ,  $snr_{i,k}$  denotes the SNR of  $ap_{i,k}$ . Therefore, the highest SNR on channel  $i$  is expressed as follows:

$$SNR_h(i) = \max\{snr_{i,1}, \dots, snr_{i,k}, \dots, snr_{i,n}\}, snr_{i,k} \in N_i, \quad (9)$$

where  $SNR_h(i)$  denotes the highest SNR value in the set of  $S_i(snr)$ . Intuitively, a higher value of  $\sum_{i=1}^N SNR_h(i)$  means that the number of APs discovered is high for this scan because the sum of  $\sum_{i=1}^N SNR_h(i)$  is proportional to the discovery rate of APs (i.e.  $\sum_{i=1}^N SNR_h(i) \propto D$ ). Therefore, for each channel  $i$ , the problem is formulated as follows:

$$\arg\{T_{Preq}(i), T_{sw}(i), t_{min}(i), t_{max}(i)\} \max \frac{\sum_{i=1}^N SNR_h(i)}{\sum_{i=1}^N T_{sc}(i)} \quad (10)$$

chromosome

subject to :

$$0 \leq T_{Preq}(i) \leq 200ms, 0 \leq T_{sw}(i) \leq 0.5ms, \quad (11)$$

$$5 \leq t_{min}(i) \leq 15ms, 3 \leq t_{max}(i) \leq 90ms, \quad (12)$$

$$t_{min}(i) \leq t_{max}(i), \quad (13)$$

where the constraints expressed in Eq.11 - 13 are the boundary values for the scanning timers as used in [9].

This formulation presented in Eq.10 is a combinatorial problem of nonlinear objective and constraint functions. In this paper, we use GA to solve this problem because GA are well suited for solving nonlinear objective and constraint functions expressed as both discrete and continuous variables [10].

#### B. Self-Optimizing: Pre-configured optimized timers using Genetic Algorithm

The proposed GA algorithm is described by Algorithm 1. The following subsections will introduce the details of each GA procedure.

**Algorithm 1** Optimization of scanning parameters based on GA

- 1:  $\alpha$ : Population size;
- 2:  $\beta$ : Elitism rate,  $\beta = 1/\alpha$ ;
- 3:  $\gamma$ : Crossover rate;
- 4:  $\theta$ : Mutation rate;
- 5:  $\delta$ : Number of generations;
- 6: Generate initial population  $Pop$ ;
- 7: **for**  $i$  to  $\delta$  **do**
- 8:   Evaluate fitness value of  $Pop$  by Equation 14;
- 9:   Select best-ranking solutions for next generation;
- 10:   Generate new children by crossover with rate  $\gamma$ ;
- 11:   Mutate the children with mutation rate  $\theta$ ;
- 12:   Select  $\alpha \times \beta$  of best solutions as elitism for next generation
- 13:   Update  $Pop$  with new population;
- 14: **end for**

1) *Chromosome representation*: The first aspect of a GA is how the solutions are encoded as chromosomes. A chromosome is a fixed-length string of genes represented for an individual. The genes can be binary or real-value numbers depending on the type of problems. The genes in a chromosome and the length of each chromosome depend on the parameters involved in the problems.

Conventionally, binary strings are used to represent the decision variables of the optimization problem in the genetic population, irrespective of the nature of the decision variables. A binary-coded GA has number of difficulties in dealing with continuous search spaces in our timer optimization problem. The use of real value-coded (RVC) GA representation has a number of advantages over binary coding. The efficiency of the GA is increased as there is no need to convert the solution variables to the binary type, less memory is required, there is no loss in precision by discretization to binary or other values, and there is greater freedom to use different genetic operators [11].

Therefore, in this paper we propose a RVC genetic algorithm in which the optimization variables are represented as the real-valued parameters. In our RVC representation, the scanning timers of all channels are considered as genes. They are  $t_{min}(i)$ ,  $t_{max}(i)$ ,  $T_{P_{req}}(i)$ ,  $T_{sw}(i)$  and  $i$  is the channel number. Each gene represents a parameter in the problem. If the number of scanning channels is  $N$ , the size of each chromosome is an array of string of length  $N * 4$ .

In Fig. 2, chromosome representation is shown. Each gene is indicated as below: S1 is denoted as  $t_{min}(1)$ , S2 is denoted as  $t_{max}(1)$ , S3 is denoted as  $T_{P_{req}}(1)$ , S4 is denoted as  $T_{sw}(1)$ , S5 is denoted as  $t_{min}(2)$  and so on.

2) *Population Initialization*: The initial population described by  $Pop$  is a randomly generated set of chromosomes. The population size in our proposed GA is  $\alpha$  (see line 1, 6 in Algorithm 1). Each chromosome is a set of scanning parameters. For example, if the number of channel is 11. For each channel  $i$ , it has four scanning parameters  $t_{min}(i)$ ,  $t_{max}(i)$ ,  $T_{P_{req}}(i)$  and  $T_{sw}(i)$ . Thus, one chromosome has

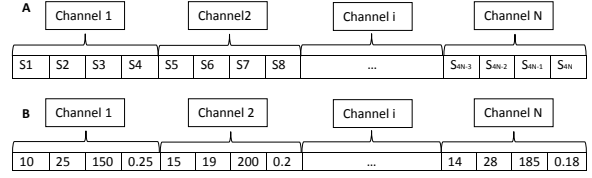


Fig. 2. Chromosome representation: (A) chromosome encoding (B) an example of chromosome

44 genes. The initial values of scanning parameters  $t_{min}(i)$ ,  $t_{max}(i)$ ,  $T_{P_{req}}(i)$  and  $T_{sw}(i)$  are randomly generated by pseudorandom numbers generators.

3) *Evaluation*: The fitness value is used to accurately evaluate the quality of the chromosomes in the population (see line 8 in Algorithm 1). The fitness value can be calculated by fitness function by Equation 14. It gives a fitness value to each chromosome. The probability that a chromosome will be selected for reproduction is based on the fitness value. The chromosomes with high fitness values have more chances to be selected for reproduction.

$$f_j = \frac{\sum_{i=1}^N SNR_h(i)}{\sum_{i=1}^N T_{sc}(i)}, \quad (14)$$

where  $f_j$  represents the fitness value of the  $j_{th}$  chromosome. We expect to use the lowest scanning time  $\sum_{i=1}^N T_{sc}(i)$  to find more available APs with higher  $SNR$  values. When  $f_j$  is the highest, we can find the chromosome with best fitness value to be selected.

4) *Selection*: There are several selection methods, such as Tournament Selection, Proportional Roulette Wheel Selection, Rank-based Roulette Wheel Selection and so on. We use tournament selection in our method, because this is more appropriate for small size problem [12]. The chromosome with the better fitness value will be selected as parents to produce new child chromosomes by the crossover operation as described in the following section. At the end of each generation, we also employ another selection method called elitism. This method selects the best chromosome in the current generation to pass to the next generation. This action is important to assure that the relevant features of the chromosomes, selected so far by the survival of the fittest principle, will not be discarded during the process of recombination (see lines 2, 9, 12 in Algorithm 1).

5) *Crossover*: The combination of two parent chromosomes produces a new child chromosome and this is called a crossover. With the real value-coded representation, the evaluation procedure and reproduction operator remain the same as that in binary-coded GA, but crossover and mutation operators for the RVC GA needs to be redefined. There are many types crossover for RVC GA such as single point crossover, n-points crossover, linear crossover, blend crossover, simulated binary crossover and so on. In the n-points crossover, according to the number of crossover points, there are also two-points, three-points and so on.

In our proposed GA method, the scanning parameters  $t_{min}(i)$ ,  $t_{max}(i)$ ,  $T_{P_{req}}(i)$  and  $T_{sw}(i)$  are considered as a

group of parameters per channel. Therefore, we choose the n-points crossover method to cut the random position based on channel numbers. We use two-points crossover because it will enable the problem space to be searched more thoroughly than one point crossover [13]. By crossover, the scanning parameters for each channels of two selected parents will be exchanged. The crossover rate is  $\gamma$  (see lines 3, 10 in Algorithm 1).

6) *Mutation*: Mutation is used to maintain genetic diversity. There are several types mutation such as Random mutation, Non-uniform Mutation and Normally distributed mutation. We choose Random mutation because it is suitable for RVC GA [14]. In the Random mutation, a random gene is selected and its value is changed within the entire parameter range. The mutation rate is  $\theta$  (see lines 4, 11 in Algorithm 1). The example is given in Fig. 3.

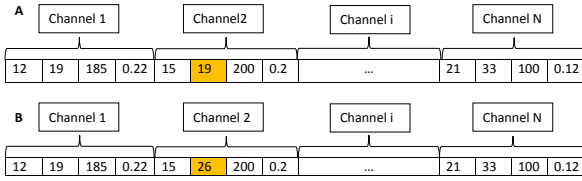


Fig. 3. Mutation: (A) before mutation (B) after mutation

7) *Termination*: The GA process can be terminated by two ways. One is when the best AP is discovered before the maximum generation is reached. When the  $SNR_h$  defined in Equation 9 of a certain AP is higher than a pre-defined maximum threshold  $SNR_{up}$ , the GA process and active scan are terminated. The current chromosomes will be selected as the best chromosome. Otherwise, the GA will continue to run until the last generation.

$$N_{gen} = \begin{cases} j, & \text{if } SNR_h \geq SNR_{up} \\ N_{max}, & \text{otherwise} \end{cases} \quad (15)$$

where  $N_{gen}$  is the generation needed to run GA procedures,  $j$  is the  $j$ th generation when  $SNR_h \geq SNR_{up}$  is satisfied before the best chromosome is found,  $N_{max}$  is the maximum number of generation size.

8) *Sorting selected channels*: To provide seamless handover for real-time applications, reducing scan time for each channels is not enough. The total scanning time is affected by both the probe waiting time and channel switching time when  $T_{P_{req}}(i)$  is fixed, described in Equation 2. Thus, after GA process, the channel will be sorted according to the value of  $SNR_h(i)$ .

There are two threshold, a pre-defined maximum threshold  $SNR_{up}$  and a pre-defined minimum threshold  $SNR_{low}$ . The channel will be ranked and selected between two threshold  $SNR_{low}$  and  $SNR_{up}$ . For the active scan, only the selected channel will be scan by a STA. The reduction of the number of channels will shorten the scan time. At the same time, the ordered channel list will help to find best candidate AP faster than original scan sequence.

Let  $C(ch)$  denote the set of scanning channels whereby each channel is indexed by  $i$  such that  $1 \leq i \leq n$  and  $i \in \mathbb{Z}^+$ . Using a set notation we express the channel set as:

$$C_i(ch) = \{ch_1, ..ch_k, ...ch_n\}, ch_k \in N, \quad (16)$$

where  $ch_k$  denotes the channel with index  $k$ . The update algorithm finds a natural ordered set [15] (denoted by  $C_i^o(ch)$ ) based on the three radio parameters (SNR, packet delay and channel load). The IEEE 802.11k standard provide mechanisms to retrieve these radio parameters from measurement reports such as SNR (signal to noise ratio), channel load and neighbor report. After retrieving these values from measurement reports, we sort the list of channels from best to worst potential APs for handover. These information is recorded in a table stored in the controller.

### C. Self-optimizing: Adaptive timers

After setting the scanning timers by GA, these timers are dynamically adapted using Algorithm 2.

Algorithm 2 describes the dynamic adaptation of probe timers based on GA. With the information of GA readily known, the value of the timers  $MinChannelTime$  and  $MaxChannelTime$  for the channels on the GA list is optimized to reduce the wasted channel probing time (lines 12 - 15, 17 - 20 in Algorithm 2), because we already have the sorted channel information.

---

#### Algorithm 2 Dynamic adaptation of $MinChannelTime$ and $MaxChannelTime$

---

```

1:  $Channel(i)$ : channel  $i$ ;
2:  $t_{min(i)}$ :  $MinChannelTime$  of channel  $i$ ;
3:  $t_{max(i)}$ :  $MaxChannelTime$  of channel  $i$ ;
4:  $t_{P_{req}}(i)$ : Probe request interval  $T_{P_{req}}(i)$ ;
5:  $t_{sw(i)}$ : Channel switching time  $T_{sw}(i)$ ;
6:  $t_{min(i)_{new}}$ : Optimized  $MinChannelTime$ ;
7:  $t_{max(i)_{new}}$ : Optimized  $MaxChannelTime$ ;
8:  $t_{P_{req}(i)_{ga}}$ : GA Optimized probe request interval;
9:  $t_{sw(i)_{ga}}$ : GA Optimized channel switching time;
10: for GA selected sorting  $Channel(i)$  do
11:   if AP is found then
12:      $t_{min(i)} \leftarrow \min\{0, t_{min(i)_{new}} - \alpha\}$ ;
13:      $t_{max(i)} \leftarrow \min\{0, t_{max(i)_{new}} - \beta\}$ ;
14:      $t_{P_{req}(i)} = t_{P_{req}(i)_{ga}}$ ;
15:      $t_{sw(i)} = t_{sw(i)_{ga}}$ ;
16:   else
17:      $t_{min(i)} \leftarrow \min\{t_{min(i)_{new}} + \alpha, \gamma\}$ ;
18:      $t_{max(i)} \leftarrow \min\{t_{max(i)_{new}} + \beta, \gamma\}$ ;
19:      $t_{P_{req}(i)} = t_{P_{req}(i)_{ga}}$ ;
20:      $t_{sw(i)} = t_{sw(i)_{ga}}$ ;
21:   end if
22: end for
```

---

When new APs are discovered in a scan, its timers are reduced by  $\alpha$  and  $\beta$  respectively to avoid excessive waiting time (see lines 12 - 15 in Algorithm 2). For existing APs in the list that have not replied to a probe request, the timers are

increased to avoid missing out potential APs (see lines 17 - 20 in Algorithm 2). The values of  $\alpha$  and  $\beta$  are the step increments for the timers. In this paper these values are constant,  $\gamma$  is the maximum allowed scanning time specified by the 802.11 standard (12ms). After the scanning phase, we obtain the best AP for reassociation. The complete proposed handover process is illustrated Fig. 4.

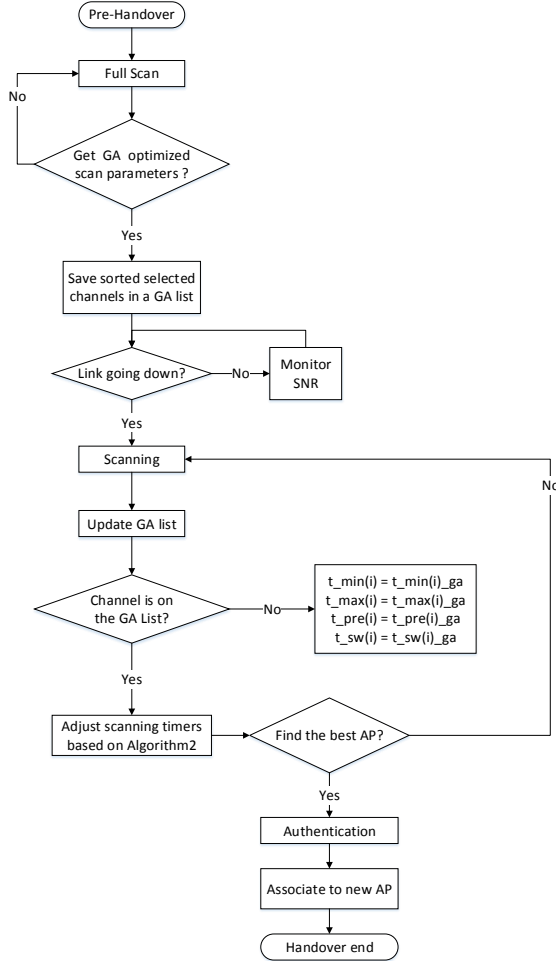


Fig. 4. Flowchart of SON handover scheme

#### IV. PERFORMANCE EVALUATION

In this section, the performance of the proposed SON method is evaluated compared with 802.11 standard using the fixed timers suggested in [1] and our previous neighbor list mechanism (NLM) handover scheme [16].

##### A. Simulation Setup

We implemented the proposed algorithms in ns-3 with all AP operating in the 2.412 GHz band (Channel 11) using the IEEE 802.11k protocol. The values of simulation parameters are listed in Table I. We implemented and collected the handover delay, throughput and packet loss rate over 30 running times with a confidence interval of 95% under the assumption that the averages are normally distributed.

TABLE I  
SIMULATION PARAMETERS

| Parameter                        | Value        |
|----------------------------------|--------------|
| Total simulation time (t)        | 30000s       |
| Speed                            | 1m/s - 10m/s |
| Number of AP                     | 6 - 30       |
| Distance between two APs         | 30 m         |
| Number of Mobile Nodes           | 1-35         |
| ActiveProbing                    | true         |
| WiFi Standard                    | 802.11k      |
| PacketSize                       | 10000 byte   |
| DataRate                         | 1 Mbps       |
| GA population size               | 50           |
| GA Maximum number of generations | 2000         |
| GA Crossover rate                | 0.8          |
| GA Mutation rate                 | 0.15         |
| GA Elitism rate                  | 0.02         |

##### B. Analysis of Results

In the simulation, we simulated three different scenarios. The first two scenarios involve six APs and five STAs. The APs are placed on a line and each AP is separated from its neighbors by 30 meters. These scenarios reflect a typical indoor office or campus WLAN.

In the first scenario (Scenario I), we want to explore how the handover delay changes with increasing STAs speed. We simulated various STAs speed from 1m/s to 6m/s. At the simulation start time  $t = 0$ , the STAs are on the same location and are connected to the same AP number 1. All STAs start to move at simulation time  $t = 0 + \delta$  where  $\delta$  is a random jitter below 0.1s.

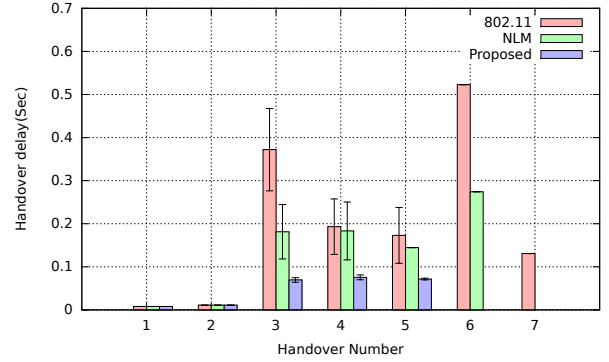


Fig. 5. Handover delay vs. handover number (Scenario I)

Figure 5 shows the average handover delay impacted by varying STAs speed. The handover number certainly increases with the moving speeds increased. However, compared with standard 802.11 and NLM, the average handover numbers of the proposed SON approach is reduced to 5. The standard 802.11 and NLM method shows 6 and 7 handover numbers for 802.11 and NLM respectively on average.

The standard layer 2 handover scheme exhibits a lower performance, with an average handover delay of 0.33s/ho (handover numbers). We can see the average handover delay of the proposed method is 0.07s/ho and it has been reduced by 78.79% compare with standard 802.11. Although the NLM improves to 0.19s/ho, our proposed SON method achieves a better handover performance than NLM because the number

of scanned channels and scanning timers are both optimized in SON, while NLM only reduces the set of scanning channels.

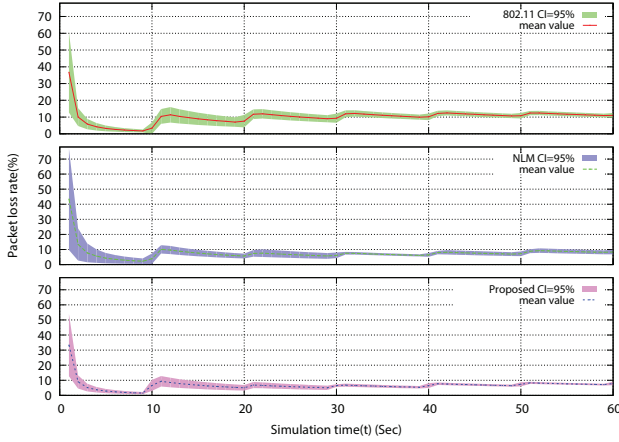


Fig. 6. Packet loss rate vs. simulation time (Scenario II)

In the second scenario (Scenario II), the STAs move at a constant speed but in random directions. The STAs are at different locations and are connected to different APs when they start to move. When a handover occurs, it usually causes packets loss due to the momentary loss in connectivity. Fig. 6 illustrates the impact of handover on the data reception in the three methods: 802.11, NLM and the proposed SON scheme. These three packet loss rates are 10.10%, 7.70% and 6.73% respectively on average. We find that the proposed SON achieves average 3.37% lower packet loss rate compared with the standard 802.11 mechanism. With the reduction of channel numbers in NLM, the packet loss rates have already been reduced. However, the packet loss rate of our proposed SON drops by 12.60% compared to NLM because the self-optimizing timers further reduce the scanning delay which in turn provides more responsive handovers.

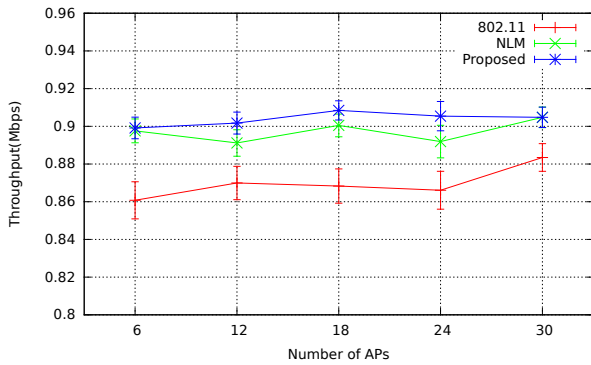


Fig. 7. Throughput vs. number of APs (Scenario III)

In Scenario III, we simulate the STAs moving between different number of APs (6, 12, 18, 24, 30) at a constant speed of 6m/s. With the increasing number of APs, the handover frequencies increase and the STAs have more chances to find new APs that can provide better throughput than the current serving AP. However, the expected communication time be-

tween the STAs and APs is reduced because of the dense environment. Because the proposed SON method considers the varying wireless network conditions and dynamically adjusted scanning timers and scanning channels, the throughput is less affected than 802.11 and NLM. Fig. 7 gives the average throughput among 802.11, NLM and the proposed SON method. The average throughput of proposed SON method has been improved by 3.93% and 0.75% compared with IEEE 802.11 standard and NLM respectively.

### C. GA performance evaluation

Figures 8 and 9 shows the convergence of the GA. The X axis describes the number of generations (iterations). The Y axis shows the best fitness value calculated by Equation 14. We aim to find the chromosome with best fitness value to be selected. Therefore, the higher of the fitness value is better.

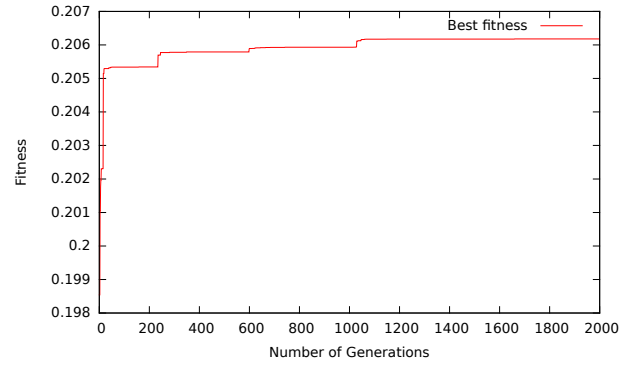


Fig. 8. The best fitness of the GA with increasing generations

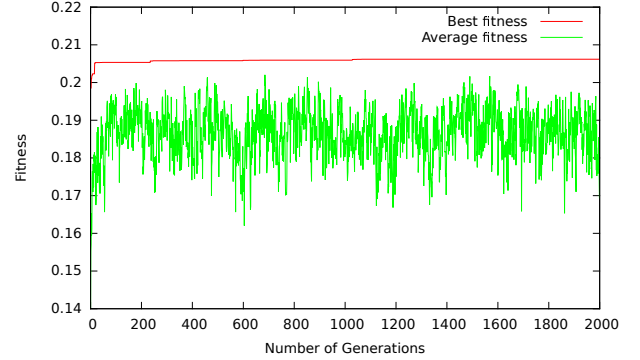


Fig. 9. The mean fitness and best fitness of the GA

As seen in Fig. 8, with the number of generations increases, the trend of best fitness value goes higher. At the beginning, the best fitness increases rapidly and then plateaus until 50 generations. The GA continues to search for the best fitness values and finds the best fitness after 1048 generations (fitness value of 0.206). This rapid increase and plateau occurs because that the algorithm finds better and better solutions that are harder to improve upon.

We compare the average fitness and best fitness in Fig. 9. The average fitness is the mean fitness values across the entire

population. In each generation, the population changes and will get a new average population fitness. The best fitness tends to improve as the iterations increase and the average fitness is always inferior to the best fitness, but the difference between them narrows after 30 generations.

## V. CONCLUSION

This paper proposed the GA method for self-optimizing of scanning parameters to improve handover in IEEE 802.11 WLANs. The simulation results demonstrate that the handover performance significantly improves compared with a traditional 802.11 scanning method using the fixed scanning timers and the number of channels scanned. In the future, we will continue to research optimizing the scanning parameters considering the effect of STAs speed and the application requirements.

## REFERENCES

- [1] A. Mishra, M. Shin, and W. Arbaugh, "An empirical analysis of the ieee 802.11 mac layer handoff process," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 2, pp. 93–102, 2003.
- [2] S. Shin, A. G. Forte, A. S. Rawat, and H. Schulzrinne, "Reducing mac layer handoff latency in ieee 802.11 wireless lans," in *Proceedings of the second international workshop on Mobility management & wireless access protocols*. ACM, 2004, pp. 19–26.
- [3] A. Zubow, S. Zehl, and A. Wolisz, "Bigap seamless handover in high performance enterprise ieee 802.11 networks," in *Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP*. IEEE, 2016, pp. 445–453.
- [4] H. Wu, K. Tan, Y. Zhang, and Q. Zhang, "Proactive scan: Fast handoff with smart triggers for 802.11 wireless lan," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications*. IEEE, 2007, pp. 749–757.
- [5] R. W. Pazzi, Z. Zhang, and A. Boukerche, "Performance evaluation of a fast mac handoff scheme using dynamic adjustment of scanning parameters," in *Proceedings of the 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*. ACM, 2009, pp. 346–352.
- [6] A. Boukerche, Z. Zhang, and R. W. Pazzi, "A self-configured handoff scheme for ieee 802.11-based wireless networks," in *Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on*. IEEE, 2009, pp. 124–129.
- [7] G. Castignani, A. Arcia, and N. Montavont, "A study of the discovery process in 802.11 networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 15, no. 1, pp. 25–36, 2011.
- [8] H. Velayos and G. Karlsson, "Techniques to reduce the ieee 802.11 b handoff time," in *Communications, 2004 IEEE International Conference on*, vol. 7. IEEE, 2004, pp. 3844–3848.
- [9] A. Arcia-Moret, A. Araujo, J. Aguilarz, L. Molinax, and A. Sathiseelan, "Intelligent network discovery for next generation community wireless networks," in *Wireless On-demand Network Systems and Services (WONS), 2016 12th Annual Conference on*. IEEE, 2016, pp. 1–7.
- [10] U. Mehboob, J. Qadir, S. Ali, and A. Vasilakos, "Genetic algorithms in wireless networking: techniques, applications, and issues," *Soft Computing*, vol. 20, no. 6, pp. 2467–2501, 2016.
- [11] D. Devaraj and B. Yegnanarayana, "Genetic-algorithm-based optimal power flow for security enhancement," *IEE Proceedings-Generation, Transmission and Distribution*, vol. 152, no. 6, pp. 899–905, 2005.
- [12] N. M. Razali, J. Geraghty *et al.*, "Genetic algorithm performance with different selection strategies in solving tsp," in *Proceedings of the world congress on engineering*, vol. 2. International Association of Engineers Hong Kong, 2011, pp. 1134–1139.
- [13] S. Picek and M. Golub, "Comparison of a crossover operator in binary-coded genetic algorithms," *WSEAS transactions on computers*, vol. 9, pp. 1064–1073, 2010.
- [14] S. Sivanandam and S. Deepa, *Introduction to genetic algorithms*. Springer Science & Business Media, 2007.
- [15] U. Faigle and G. Turán, "Sorting and recognition problems for ordered sets," *SIAM Journal on Computing*, vol. 17, no. 1, pp. 100–113, 1988.
- [16] L. Hao, B. Ng, and Y. Qu, "Dynamic optimization of neighbor list to reduce changeover latency for wi-fi networks," in *Proceedings of the 2017 International Conference on Telecommunications and Communication Engineering*. ACM, 2017, pp. 20–24.